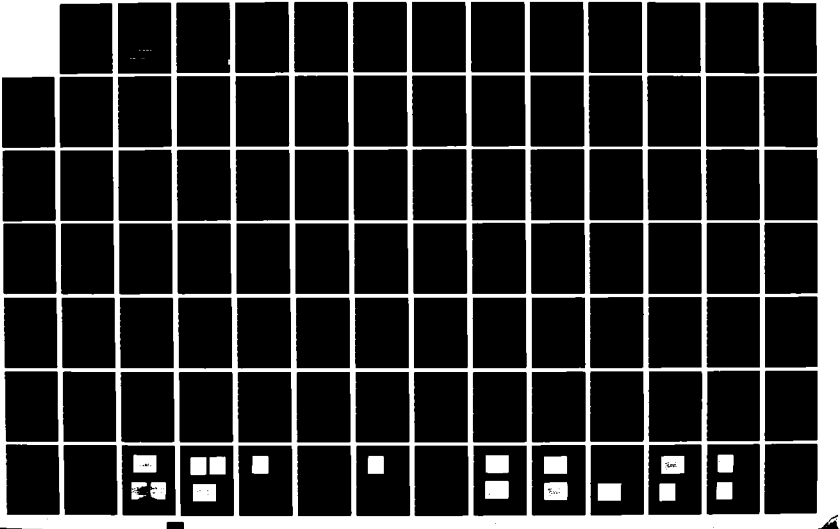
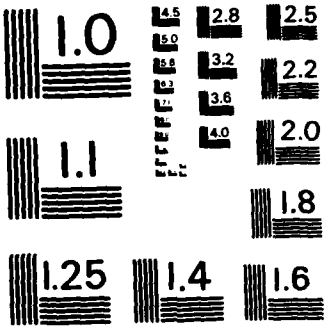
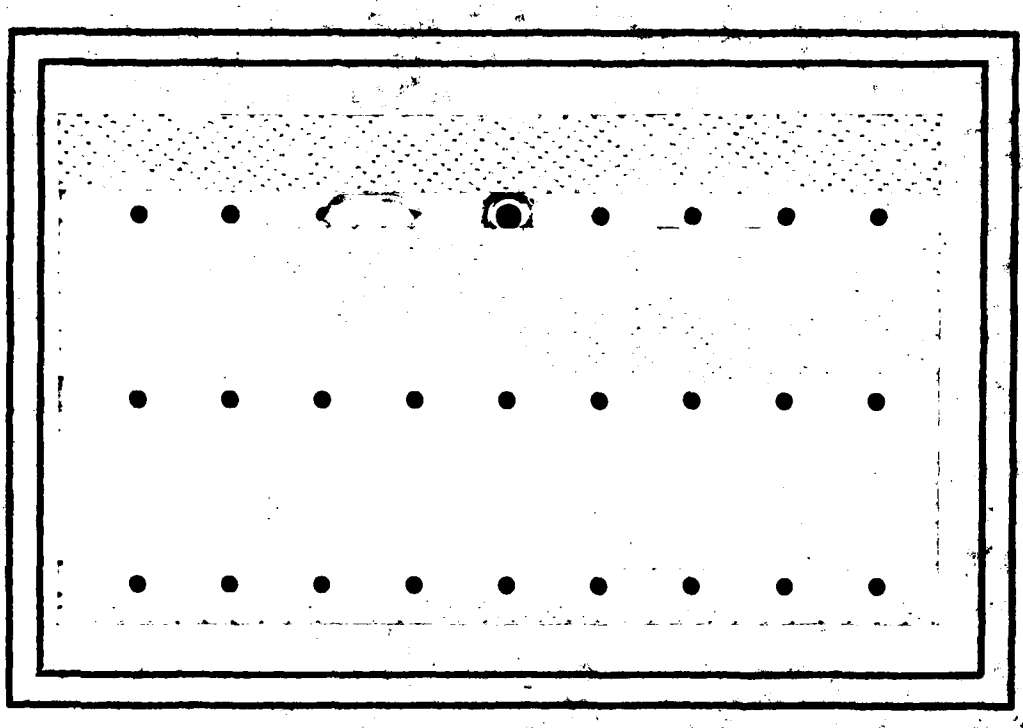
AD-A160 129 HYPOTHESIS INTEGRATION IN IMAGE UNDERSTANDING SYSTEMS (U) MARYLAND UNIV COLLEGE PARK CENTER FOR AUTOMATION RESEARCH Y S HWANG ET AL. JUN 85 CAR-TR-130 1/2

UNCLASSIFIED AFOSR-TR-85-0876 F49620-83-C-0082 F/G 14/5 NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS - 1963 - A



COMPUTER VISION LABORATORY

CENTER FOR AUTOMATION RESEARCH

UNIVERSITY OF MARYLAND
COLLEGE PARK, MARYLAND
20742

AD-A160 129

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION: UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT: Approved for public release; distribution unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S): AFOSR-TR- 85-0876
6a. NAME OF PERFORMING ORGANIZATION: University of Maryland; 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION: Air Force Office of Scientific Research
6c. ADDRESS (City, State and ZIP Code): College Park, MD 20742	7b. ADDRESS (City, State and ZIP Code): Directorate of Mathematical & Information Sciences, Bolling AFB DC 20332
8a. NAME OF FUNDING/SPONSORING ORGANIZATION: AFOSR; 8b. OFFICE SYMBOL (If applicable): NM	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER: F49620-83-C-0082
8c. ADDRESS (City, State and ZIP Code): Bolling AFB DC 20332	10. SOURCE OF FUNDING NOS.

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT NO.
61102F	2304	K2	

11. TITLE (Include Security Classification): Hypothesis Integration in Image Understanding Systems

12. PERSONAL AUTHOR(S): Vincent Shang-Shouq Hwang, Larry S. Davis and Takashi Matsuyama

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Yr., Mo., Day)	15. PAGE COUNT
Interim	FROM _____ TO _____	June 1985	97

16. SUPPLEMENTARY NOTATION

17. COSATI CODES		
FIELD	GROUP	SUB GR.
XXX	XXXXXXXXXXXXX	

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number): image understanding systems, SIGMA, robust control strategy

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

The goal of this research is to develop a robust control strategy for constructing image understanding systems (IUS). This paper proposes a general framework based on the integration of "related" hypotheses. Hypotheses are regarded as predictions of the occurences of objects in the image. Related hypotheses are clustered together. A "composite hypothesis" is computed for each cluster. The goal of the IUS is to verify the hypotheses. We constructed an image understanding system, SIGMA, based on this framework and demonstrated its performance on an aerial image of a suburban housing development.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION
UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS ☐	UNCLASSIFIED

22a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE NUMBER (Include Area Code)	22c. OFFICE SYMBOL
Robert N. Buchal	(202) 767-4939	NM

DD FORM 1473, 83 APR EDITION OF 1 JAN 73 IS OBSOLETE

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

CAR-TR-130
CS-TR-1513

F49620-83-C-0082
June 1985

HYPOTHESIS INTEGRATION IN IMAGE UNDERSTANDING SYSTEMS

Vincent Shang-Shouq Hwang
Larry S. Davis
Takashi Matsuyama*

Center for Automation Research
University of Maryland
College Park, MD 20742

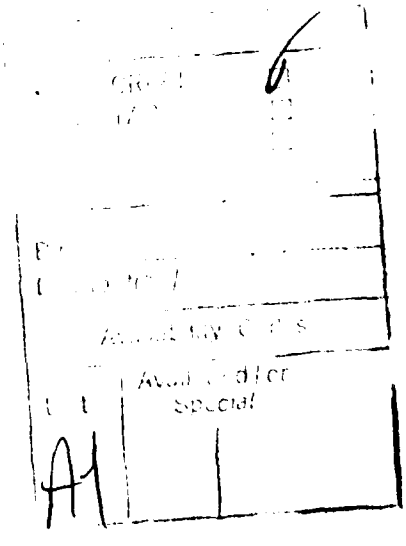
ABSTRACT

The goal of this research is to develop a robust control strategy for constructing image understanding systems (IUS). This paper proposes a general framework based on the integration of "related" hypotheses. Hypotheses are regarded as predictions of the occurrences of objects in the image. Related hypotheses are clustered together. A "composite hypothesis" is computed for each cluster. The goal of the IUS is to verify the hypotheses. We constructed an image understanding system, SIGMA, based on this framework and demonstrated its performance on an aerial image of a suburban housing development.

The support of the U.S. Air Force Office of Scientific Research under Contract F49620-83-C-0082 and of the National Aeronautics and Space Administration under Grant NASA9-16664 are gratefully acknowledged.

*Permanent address : Department of Electrical Engineering, Kyoto University, Sakyo, Kyoto, Japan.

Table of Contents



1. Introduction

A primary objective in computer vision research is to construct image understanding systems (IUS's) which can analyze images based on object models. Usually, an IUS analyzes images by constructing *interpretations* in terms of the object models given to the IUS. *Interpretation* refers to the mapping between objects (e.g., houses, roads) in the object model and image structures (e.g., regions, lines, points) in the image. During the analysis, an IUS needs to perform the following two types of tasks:

- segmentation : the task of grouping pixels together to construct image structures that can be associated with objects in the given model.

- interpretation : the task of constructing mappings between image structures and objects.

Segmentation is practical when sufficient knowledge is available about the image to be processed and the image structures to be computed. The base of knowledge increases as the interpretation process develops, leading to more constrained and therefore more reliable segmentation.

Many IUS's were constructed in the late 1970's ([Barr81], [Ball82], [Binf82] [Ball82].) Most systems integrate segmentation and interpretation using one of the following types of analysis.

1) Bottom-up analysis: the image structures are extracted from the image, and are interpreted as instances of the objects in the model. For example, when a large rectangular region is extracted, interpret it as a house.

2) Top-down analysis: the appearance of the object is first determined, and the associated image structures are extracted. For example, suppose an IUS wants to find a house; the IUS invokes the house model and establishes the descriptions of the specific image structures to be extracted from the image.

It is generally accepted that image understanding systems should incorporate both bottom-up and top-down analyses. Some systems use only one type of analysis. MSYS [Barr76] developed by Barrow and Tenenbaum used bottom-up analysis. Image structures are first segmented from the image. A set of initial labels are assigned to these image structures (based on height, homogeneity, etc.) Then, geometric constraints between labels are used to filter out inconsistent labelings. Bolles [Boll76], on the other hand, used top-down analysis. In his system , a goal is first constructed. The system then matches the goal, which is represented as a template, with the image. A similar approach is used in Garvey's [Garv76] system. Other systems (Hanson, Riseman [Hans78]: Matsuyama [Naga80]) incorporate both types of analysis but use ad hoc rules to determine which type of analysis is to be used at what stage during the analysis. Such systems often require a large set of domain dependent control knowledge to direct the analysis of the IUS.

It is the goal of this research to develop a robust control strategy for constructing image understanding systems, thus eliminating the need to use large amounts of domain specific control knowledge in specific applications. In this paper, we propose a general framework which enables IUS's to integrate both bottom-up and top-down analyses into a single flexible reasoning process. We construct an image understanding system, SIGMA, based on this framework and provide demonstrations of its performance on images of a suburban housing development.

1.1. Integration of hypotheses

Considering the following proposition:

> If a structure of type x is present in the scene having certain spatial properties, then there should exist a structure of type y having certain properties in the image.

It is often the case that what is known about x is not sufficient to completely characterize y (i.e., we might be able to predict its size and color, but perhaps not its orientation). In addition, there might be many x's, each predicting the occurrence of y, but each contributing different constraints on the properties of y.

For example, by locating a house in the image, one may predict the occurrences of other objects, e.g., neighboring houses. Furthermore, the discovery of a rectangular homogeneous region in the image may also generate

a prediction of a house. It is usually the case (depending on the object model) that each of these predictions provides some "cues" about the occurrence of a house and it is the integration of all these cues that may characterizes the occurrence of a house adequately enough to easily recognize it.

Let us call the predictions about the occurrences of objects in the image *hypotheses*. Suppose several hypotheses, which may be independently generated, are predictions about objects at the same location in the image. It is reasonable to assume that these hypotheses are predictions about the "same" object, although each may only constrain some subset of the properties of the object. By integrating these hypotheses, an IUS could construct a more complete description of the object and use it to direct a more effective and informed analysis.

1.2. An overview of the SIGMA image understanding system

Figure 1-2 shows the system architecture of the SIGMA image understanding system. The user provides object models to SIGMA, and the results of the analysis are available to the user through a query-answering module.

The image is first segmented by a general purpose low level vision system (LLVS). The segmentation results are recorded in the iconic/symbolic database. The high level vision system (HLVS) uses the object model either to interpret image structures already extracted or to direct the low level processes to search for image structures not yet discovered. During the

analysis, the HLVS incrementally constructs an interpretation network for the input image. A "goal" is given to the query-answering module (QAM). At the end of each analysis iteration, the QAM is activated and "matches" the current status of the analysis with the goal. This construction process continues until the "goal" is accomplished (i.e., a successful match between the current status of the analysis and the goal) or no more interpretations can be constructed. At this stage, the QAM provides the current status of the analysis. In the following subsections, we present each module of SIGMA in more detail.

1.2.1. The low level vision system

In SIGMA, the LLVS is formulated as a domain-independent goal-directed segmentation system. A goal, which is described by a list of constraints on the image structures to be computed, is given to the LLVS. The LLVS uses general segmentation techniques to extract such image structures. Other systems have been constructed to perform goal-directed segmentation - e.g., Selfridge [Self82] and Nazif & Levine [Nazi84].

Our approach differs from the approaches taken in these systems. We assume that many specialized methods are needed to extract image features from the image. An LLVS needs to select, from a pool, methods that best suit the task. Furthermore, new methods are frequently developed that can augment or replace the methods currently used by the LLVS. It is important to

design an LLVS so that adding methods to it is easy.

Our LLVS is based on a select-and-schedule strategy. When the LLVS is asked to verify some hypothesis, it first selects those methods which are applicable by matching the hypothesis against a decision table. Then, the LLVS schedules the selected methods according to their potential. If one method fails to verify the hypothesis, the next method will be tried until the hypothesis is verified or until all methods have been tried and have failed. This approach is similar to the "blackboard" method [Davi77] and the "contract net" idea [Smit78]; but the implementation here is simpler. For a detailed discussion of the LLVS, see [Hwan84].

1.2.2. The high level vision system

The high level vision system (HLVS) uses object models to interpret data recorded in the iconic/symbolic database and construct an interpretation network. The HLVS uses the integration of hypotheses principle to direct analysis. This is implemented by the following reasoning steps.

1) Hypothesis generation: the HLVS generates hypotheses about occurrences of objects in the image.

2) Hypothesis integration: the HLVS clusters "related" hypotheses together.

3) Hypothesis abstraction: the HLVS computes a "composite hypothesis" for each cluster.

4) Hypothesis verification: the HLVS selects hypotheses and verifies them by computing values for those attributes which are not completely

constrained.

The HLVS performs the reasoning iteratively. At the end of each iteration, the HLVS checks whether the "goal" is accomplished by activating the QAM. If the goal is accomplished or no more interpretations can be constructed, the construction process terminates and the status of the analysis is available through the QAM.

1.2.3. Query-answering module

Potentially, SIGMA constructs all possible interpretations for an image. However, SIGMA needs to select, among many interpretations, a good one as its conclusion. Instead of finding a "best interpretation", we model this selection process as a database query answering process. A program (QAM) was developed to answer simple queries about the interpretation network and to display the associated image structures.

The goal of the analysis is provided to the QAM as a query. Whenever the QAM is activated (by the HLVS), it matches the goal with the interpretations already constructed. If any interpretation that satisfies the goal is found, the QAM enters into an answer mode and provides a query-answering capability for selecting "good interpretations" and displaying the explanations for these interpretations.

1.3. Outline of the paper

We first present the knowledge representation paradigm used in SIGMA. In Section 3, we discuss a framework for performing hypothesis integration and abstraction. This is followed by a detailed description of the system constructed based on this framework. Conclusions are presented in the final section.

2. Representation of object models

2.1. What to represent?

The knowledge representation formalism determines a general framework for organizing the necessary knowledge into a knowledge base and supports a powerful inference mechanism for guiding the recognition of a specific scene. An appropriate knowledge representation tool can often simplify the task of transferring problem domain expert knowledge into knowledge bases in computer systems.

Consider the following house model:

> A house is "rectangular" or "L-shaped"; its area is larger than 1000 square feet but no larger than 2500 square feet. A house usually belongs to a group of houses which are on the same side of a road. Roads can be found near the house. Usually, the road is parallel or perpendicular to the house and a driveway connects the road to the house.

Based on how an IUS uses such a model to locate houses in a given image, one can categorize this scene knowledge into the following classes.

1) *What to look for.* This class of knowledge describes the appearances of objects (e.g., the type of image structures associated with objects.) In the house example, the appearance of the house is a homogeneous compact rectangular region. To locate houses, an IUS segments the input image and identifies as houses those regions which are rectangular and compact and whose sizes are between 1000 and 2500 square feet.

2) *Where to look.* This class of knowledge includes the geometric and topological relations between objects. The knowledge base might, for example, specify (based on connectivity, relative orientation, etc.) relations between

'riveways, houses, and roads. An IUS might, if one of these objects is discovered (say a driveway), use this relation to initiate and constrain the search for other objects (e.g., a connected house and road) not yet discovered. An IUS might also use such relations to examine whether a house, a driveway, or a road already discovered satisfy the required relations.

3) *When to look.* This class of knowledge describes strategies regarding the application and confirmation of relations. One the one hand, we often want to postpone applying a specific piece of relational knowledge until sufficient information has been obtained to strongly suggest that the relation may be applicable. On the other hand, since the confirmation process often involves the searching of image structures associated with other objects, we might also want to postpone the confirmation of a specific relation until a sufficient description of the object to be searched is collected. For example, when the IUS generates a house hypothesis, instead of searching for an image structure associated with it immediately, the IUS might postpone the search until a sufficient description of the house (e.g., shape, intensity, etc.) is available.

A principal objective of this research is to develop a representation scheme which simplifies the task of capturing domain knowledge as a knowledge base for IUS's. This section presents the knowledge representation scheme used in the SIGMA system. Note that the scene model is used mainly by the HLVS (High Level Vision System) module in SIGMA.

2.2. Basic representation primitives

Our representation formalism is based on frame system theory [Mins75], semantic networks [Wino75] [Hend79], and an object oriented problem solving style [Stee79] [Wein80] [Gold83]. In SIGMA, object models are represented as a graph structure of nodes and arcs. Objects are described by "frames" (nodes in the graph structure) while relations between these objects are described by "rules" and "links" (arcs in the graph structure). In such a formalism, domain

knowledge is built around a set of objects and a set of operations that can be applied to them.

The basic entities of the representation are called *frames* and are used to model abstract objects in the problem domain such as "house" or "road". Each frame may have many associated descriptions that are defined by *slots*. Slots are similar to "property lists" in LISP. Each slot is a list which contains an indicator (i.e., name) and a value.

In addition to slots where values are recorded, we can also associate with frames all the knowledge which is used to compute values of slots. We represent this type of knowledge as *rules*.

Rules used in this context are procedural--i.e., the knowledge about how to compute values of slots is encoded in programs. As mentioned above, these "programs" are written using an object-oriented programming style.

Objects in the scene domain are often structured into hierarchies. It is often natural and convenient to preserve these hierarchies when we construct the scene model. *Links* are used to describe the hierarchical relations between objects.

One object hierarchy often used is the generalization/specialization hierarchy; CAN-BE and AKO links are employed to describe it. Link CAN-BE describes a frame and its specializations while link AKO describes a frame and its generalizations.

Properties are inherited through the AKO link. This usage is similar to the "property inheritance" in semantic networks ([Moor79], [Nils80].) All the knowledge recorded in frames that are linked to a father frame by the AKO link is inherited by that frame. For example, both the RECTANGULAR-HOUSE and the L-SHAPED-HOUSE have centroid, shape-description, front-of-house, and connecting-driveway slots. Also, both the RECTANGULAR-HOUSE and the L-SHAPED-HOUSE can use rule $F_{driveway}$ to compute the connecting driveway.

Often, the HLVS needs to reason across the CAN-BE link. For example, suppose the HLVS needs to compute the shape of a house. The HLVS is not able to do the computation since there is no such rule recorded in the HOUSE frame. Instead, the HLVS needs to reason about what specialization to choose, i.e., RECTANGULAR-HOUSE or L-SHAPED-HOUSE. The strategies for this type of reasoning are called *specialization strategies* and are encoded as rules and recorded in frames. Attaching such search strategies using CAN-BE links is similar to the process of "plan elaboration" in Garvey's system [Garv76]

As an example, suppose that there are two type of houses, rectangular and L-shaped, in community A. Every house has a driveway. However, each type of house has a different appearance. Suppose $F_{rectangle}$ is a rule which computes the shape description of a rectangular house, and $F_{driveway}$ is another rule which finds the driveway connecting to a rectangular house. Rule $F_{driveway}$ computes the driveway of a house. We can write the house model as

shown in Figure 2-1. In this model, the HOUSE frame is a generalization of the L-SHAPED-HOUSE frame and the RECTANGULAR-HOUSE frame while the L-SHAPED-HOUSE frame and RECTANGULAR-HOUSE frame are specializations of the HOUSE frame. Their hierarchical relations are shown in Figure 2-2.

2.3. Instantiation of a frame

Frames are the prototypes of objects. The SIGMA system uses frames as models to construct interpretations of the image by making instances of frames. An *instance* is a copy of a frame. The process of making instances is called *instantiation.* At instantiation, values can be assigned to slots. These values may be the "defaults" (specified in the frame definition) or may be computed using rules. Since all instances are recorded in the iconic/symbolic database in the HLVS as basic database entities, we use the term *Database Entities (DE's)* interchangeably with the term "instances" in the rest of the paper.

An important property of an object is its appearance. During the analysis, the HLVS needs to direct the LLVS (Low Level Vision System) to process the image and locate image structures which are associated with objects. Some objects' appearances are defined in terms of image structures that can be directly computed by the LLVS. Those frames which define such objects are called *primitive frames.* Frames which are not primitive are called

non-primitive frames.

Depending on what is known about the appearance of an instance, an instance can be in one of the following two states: *verified,* which indicates that the appearance of the instance is some already located image structure or is a function of the appearances of verified instances; and *hypothetical,* which indicates that the appearance of the instance has not been determined.

In addition to the appearances of objects, the HLVS also uses the iconic description of a frame during its reasoning. The iconic description specifies an area in the image and its definition is specified by a rule. During the hypotheses integration, the HLVS uses the iconic descriptions to reason whether two DE's are related (explained in Section 3). The use of iconic description in SIGMA is similar to the use of "functional areas" in Mckeown's SPAM aerial interpretation system [McKe84].

The values recorded in instances may be updated during the analysis. Every instance has a special numerical value which is called the *strength* of the instance. The method used to compute strength is described as a procedure, say $P_{strength}$, in the frame's definition. Upon instantiation, a strength is computed for each instance. Whenever the values recorded in an instance are updated, the strength of the instance is also recomputed by reevaluating $P_{strength}$. The HLVS uses such values to control its focus of attention mechanism.

Suppose one defines the appearance of a house (house frame) as a rectangular compact region and a row of houses (house-group frame) as the union of the appearances of all the houses in a house-group. Then the house frame is primitive while the house-group frame is non-primitive. In SIGMA, in order to locate a house-group, the HLVS first generates hypotheses about the location of member houses and then direct the LLVS to locate each house individually.

Now, suppose that the LLVS located a rectangular compact region, R_0. The HLVS will generate a house instance, H_1, whose appearance is R_0 and mark it as a verified instance. However, suppose the HLVS further generates neighboring house predictions for H_1, say H_2 and H_3. Both H_2 and H_3 are hypothetical instances since the appearances of these instances have not yet been determined from the image.

2.4. Representing relations between objects

A major portion of the scene domain knowledge involves relations between objects. However, these relations must be represented in forms that can be directly used by the HLVS. Our approach is influenced by production rules [Davi77] and the planning paradigm used in Garvey's vision system [Garv76].

Suppose we have the following house-road relation:

A road $road_0$ is *along* a house $house_0$ if the predicate along ($road_0$,$house_0$) is true.

There are at least two potential uses of this relation by the HLVS:

- HLVS uses the relation to check whether road $road_0$ is along house $house_0$.

- HLVS uses the relation to direct a search for a road along house $house_0$.

In order to support multiple uses of a relation by the HLVS, we use a test-hypothesize-and-act strategy to describe relations. A binary relation REL(O_1,O_2) between objects O_1 and O_2 is represented using two functional descriptions:

$$O_1 = F(O_2) \text{ and } O_2 = G(O_1).$$

Program F computes the object expected by object O_2 and is recorded in object frame O_2 as a rule. Program G computes the object expected by object O_1 and is recorded in object frame O_1 as a rule also.

As noted earlier, control knowledge for the use of relations and control knowledge for directing search are both required by the HLVS. We represent such knowledge as predicates associated with rules.

We present our rule representation scheme as follows:

A rule is composed of three parts:

<control-condition>
<hypothesis>
<action>.

<Control-condition> is a predicate. It indicates when a rule can potentially be applied. <Hypothesis> specifies the description of a desired object that is created when the <control-condition> evaluates to true. <Action> describes the code to be evaluated if <hypothesis> is verified. In general, <action> can add facts to or delete facts from the iconic/symbolic database of the HLVS.

The *house-road* relation can be written as a rule in the HOUSE frame as follows (Figure 2-3):

> To compute a road along house $house_0$, we always generate a hypothesis $road_x$ with the following slot values:
>
> road.orientation:
> greater than ($house_0$.front-of-house + 80 degrees) but less than ($house_0$.front-of-house 100 degrees).
> road.width:
> greater than ($house_0$.width * 0.3) but less than ($house_0$.width * 0.5).
> road.centroid:
> resides within REGION($house_0$.centroid + T($house_0$.front-of-house)).
>
> T(.) is a function. If the hypothesis $road_x$ is verified by some road $road_0$, then road $road_0$ is along house $house_0$.

Figure 2-4 shows a model for suburban housing developments. Objects are described by nodes (square) and relations are described by arcs. In this model, Rectangle and Picture-Boundary are the "primitive frames".

The HLVS makes use of the different parts of a rule to perform its reasoning. We discuss this in Section 4.

3. Integration of hypotheses

3.1. Introduction

Consider a binary relation REL(O_1,O_2) between two classes of objects, O_1 and O_2. This relation can be used as a constraint to recognize objects from these two classes by first extracting image structures which satisfy the specified appearances of O_1 and O_2, and then checking that the relation is satisfied by these candidate objects (Figure 3-1). In this *bottom-up* recognition scheme, analysis based on relations cannot be performed until image structures corresponding to objects are extracted.

In general, however, some of the correct image structures fail to be extracted by the initial image segmentation. So one must, additionally, incorporate *top-down control* to find image structures missed by the initial segmentation. Such top-down processes use relations to predict the locations of missing objects, as in the system described by (Garvey [Garv76], Selfridge [Self82])

As noted above, the use of relations is very different in the two analysis processes : consistency verification in bottom-up analysis and hypothesis generation in top-down analysis. An important characteristic of our hypothesis integration method is that it enables the system to integrate both bottom-up and top-down processes into a single flexible spatial reasoning process.

As will be described in Section 4, the HLVS first establishes local environments. Then, either bottom-up or top-down processes are activated depending on the nature of the local environment. The following sections describe the concepts and characteristics of this process.

3.2. The representation of database entities

All instances, hypothetical or verified, generated by the HLVS are recorded in a database. In the rest of this section, we use the term *database entity (DE)* to refer to instances recorded in the database. In addition, we use the term *hypothesis* to refer to instances in the hypothetical state.

The description of each DE consists of two parts. One part is the *iconic description.* This description is a region in the image which indicates where the DE may be located. It is generated by the rule which specifies the iconic description of the frame used to generate the DE.

The second part is the *symbolic description,* which includes the values filled into the slots of the DE, and the set of constraints imposed on these values. These constraints are represented by a set of linear inequalities in one variable (the slot name).

3.3. Consistency between a pair of DE's

"Related" DE's are integrated and analyzed together. In SIGMA, "relatedness" between DE's is defined in terms of "consistency" between pairs of DE's. A pair of DE's, DE_1 and DE_2, are said to be *consistent* if the following

conditions hold:

1) The iconic descriptions of the DE's must intersect. It is also possible to impose some requirements on the size and shape of the area of intersection.

2) The DE's are *compatible*. Let OP be the intersection arising from two DE's, and let F_1 and F_2 denote the frames from which DE_1 and DE_2 were copied. DE_1 and DE_2 are said to be *compatible* if F_1 and F_2 are linked by CAN-BE or AKO links. Otherwise, DE_1 and DE_2 are said to be *incompatible*. This will be explained in more detail in Section 3.5.

3) The constraints imposed on the attributes of the DE's must be *satisfiable*. Every DE has associated with it a set of linear inequalities in one variable that constrain the permissible values of the DE's attributes. A simple constraint manipulation system is used to check the consistency between the sets of inequalities by generating the solution space (also represented by inequalities) to the intersection of those sets. If this solution space is non-empty, then the constraints are *consistent*.

3.4. Formation of maximum consistent situations

Consistent DE's are combined into *situations*. These DE's are said to *participate* in the formation of a situation. The *P-set* of a situation is its set of participating DE's. Situation S_a is *less than* situation S_b if the P-set of S_a is a subset of the P-set of S_b. This ordering is used to structure all the situations into a *situation lattice*. Note that a single DE is also a situation. The rest of this section presents the algorithm used to form situations.

Two DE's are said to be *2-consistent* if they are consistent. In general, a set of DE's is said to be *n-consistent* if every possible subset of $(n-1)$ of the DE's is $(n-1)$-consistent. Clearly, a set of DE's is n-consistent if and only if all possible pairs of DE's in the set are 2-consistent.

When a DE, say DE_{new}, is inserted into the iconic/symbolic database, the current situation-lattice is updated by first computing the set, U, that contains all DE's whose iconic descriptions intersect with the iconic description of DE_{new}. Then, we iteratively compute all lists of n-consistent DE's for those DE's in the set U. Each such list of n-consistent DE's forms the P-set of some situation. Algorithm 3-1 describes this process.

The *maximum consistent situations* are those situations which are the roots of the situation lattice.

Algorithm 3-1 : Updating the Situation Lattice

Step 1:	Suppose the newly inserted entity is DE_{new}. Compute the set U. N=2.
Step 2:	Compute the set, R, of all the N-consistent DE's for the DE's in U. Remove any which do not contain DE_{new}.
Step 3:	If R is empty, then exit. Otherwise, insert all the elements of R into the situation-lattice.
Step 4:	Increment N by 1. Construct all the pairs for elements in R. Represent each pair by the union of the members in each element. Remove any which is not N-consistent or does not contain DE_{new}. Set R to be the set of resulting N-consistent DE's.
Step 5:	Go to step 3.

Figure 3-2 shows an example of how the situation lattice is updated when a DE is inserted. Each DE is represented by a letter. A situation is represented by all the DE's in its P-set. Figure 3-2(a) shows the situation lattice before the insertion of DE_E and the iconic descriptions of the DE's. Suppose that the new DE, DE_E, is consistent with DE_A, DE_B and DE_D. The set U would then include

$$DE_A, DE_B, DE_C, DE_D, DE_E.$$

The first time that step 3 is evaluated, set R contains the following situations:

$$DE_{AE}, DE_{BE}, DE_{DE}.$$

The second time that step 3 is evaluated, set R contains the following situation:

$$DE_{ADE}$$

The updating stops at the third iteration. Figure 3-2(b) shows the situation lattice after the updating process.

When a DE, say DE_{remove}, is being removed from the iconic/symbolic database, the current situation lattice must also be updated. This can be done simply by removing all the situations in the situation lattice which are larger than DE_{remove}.

Suppose, for example, that DE_A is removed from the situation described in Figure 3-2(b). Figure 3-3 shows the resulting situation lattice.

It is possible that the number of situations in the situation lattice may grow exponentially. In practice, this does not happen since the number of participants in a situation is usually quite small, e.g., two or three.

3.5. Constructing the composite hypothesis

A situation is a collection of consistent DE's. The HLVS selects a situation and proposes a *composite hypothesis* which "summarizes" the constraints imposed on the attributes of all the participating DE's. The strategy for computing the composite hypothesis is specified by a procedure recorded in the frame's definition. (Note that two DE's are consistent only if they are instances of the same frame or instances of frames in the same generalization/specialization hierarchy. Therefore, all the participants in a situation must be instances of frames in the same generalization/specialization hierarchy. The procedure for computing the composite hypothesis is recorded in the most general frame.) This section presents some strategies for computing the composite hypothesis.

One simple strategy is to use the solution sets of all the constraints imposed on the attributes of all the participating DE's (explained in Section 3.4) as the constraint set of the composite hypothesis. The target object of the composite hypothesis is the most specialized object expected by all the

DE's.

Suppose that the constraint set of DE_1 is

target object = HOUSE,
house.centroid = (100,130),
230 < house.area < 300

while the constraint set of DE_2 is

target object = RECTANGULAR-HOUSE,
house.centroid = (100,130),
250 < house.area < 320,
house.region-contrast > 3.

Using this method, we generate the composite hypothesis for DE_1 and DE_2 as follows:

target object = RECTANGULAR-HOUSE,
house.centroid = (100,130),
250 < house.area < 300,
house.region-contrast > 3.

Another strategy is to take the union of all the solution sets of the constraints imposed on the attributes of all the participating DE's. Suppose, for example, that two hypotheses, DE_1 and DE_2, about a road have constraints on their starting and ending points as follows:

hypothesis DE_1,
target object = road,
road.end-points = $\{(100,100),(100,150)\}$.

hypothesis DE_2,
target object = road,
road.end-points = $\{(100,125),(100,180)\}$.

We may want to construct a road hypothesis whose constraint set is the union of these constraints on DE_1 and DE_2:

target object = road,
road.end-points = $\{(100,100),(100,180)\}$.

4. An implementation of SIGMA

4.1. Overview

The goal of SIGMA is to segment the image into image structures which correspond to the objects specified in the object model. Section 1.3 outlined the architecture of the SIGMA image understanding system. This section describes its implementation.

Figure 4-1 illustrates the different stages of the control of SIGMA. SIGMA first directs the LLVS to perform an initial segmentation of the image. A set of image structures are computed at this stage. At the second stage, the HLVS constructs partial interpretations based on the results of the initial segmentation. However, during the construction, the HLVS may direct the LLVS to compute more image structures. When all construction activities finish, SIGMA provides a query-answering module for selecting "good interpretations" and displaying the reasoning paths used to derive these interpretations. During the entire analysis, SIGMA maintains a database (the iconic/symbolic database) to record all the intermediate results generated at each stage.

The rest of this section discusses the implementation of SIGMA.

4.2. Description of goals

The Query-Answering Module (QAM) is activated by the HLVS at the end of each reasoning iteration. The goal of SIGMA is described as a query to

QAM. QAM matches the query with the interpretations already constructed. If any interpretation matches the goal, QAM enters into an answer mode and provides an interactive query-answering capability.

Suppose, for example, that the goal is to locate any road whose length is longer than 300 feet in the image and has at least two houses along it. This goal can be represented by the following query:

road(x) and (x.length > 300 feet) and (x.number-of-houses >2).

During the interpretation stage, whenever a road instance is constructed whose length is longer than 300 feet and has at least two houses along it (i.e., x is bound to some interpretation constructed by the HLVS), QAM will enter an answer mode and make the specific road instance that satisfies the goal available to an interactive program. One can use this program to traverse the *interpretation network* (the network which is constructed by the HLVS during the interpretation process), and display symbolic and iconic descriptions of the interpretations constructed.

4.3. The initial segmentation

SIGMA starts its processing by directing the LLVS to extract image structures. The schematic diagram of the initial segmentation process is shown in Figure 4-2. The set, I, which contains a list of hypotheses about primitive objects, is used to describe the goal of the initial segmentation pro-

cess.

The Initial Segmentation Controller (ISC) sequentially selects hypotheses from the set I and directs the LLVS to extract image primitives which satisfy these hypotheses. For each image primitive extracted, the ISC makes an instance of the frame of which the hypothesis is a copy, and then inserts the instance created into the iconic/symbolic database.

Suppose, for example, that we want to first extract all regions which might correspond to house groups and roads in the image. A set which contains the following hypotheses can be used as the set I:

hypothesis 1: /* extract compact and bright rectangles */
target object = rectangle,
in-window = whole image,
rectangle.elongatedness $\leq$ 10,
rectangle.compactness $<$ 18,
rectangle.region-contrast $>$ 3,
180 $<$ rectangle.area-of $<$ 400.

hypothesis 2: /* extract elongated rectangles */
target object = rectangle,
in-window = whole image,
7 $<$ rectangle.width $<$ 20,
rectangle.elongatedness $>$ 10,
rectangle.length $>$ 10,
rectangle.compactness $\geq$ 18,
rectangle.region-contrast $>$ 3.

The set I for the initial segmentation could, in principle, be computed from the scene model, since the appearances of objects are described in terms of the appearances of "primitive frames". The ISC could choose those primi-

tive frames whose appearances are salient (i.e., they can be located "easily" by the LLVS) as the I-set. However, this was not implemented in SIGMA; the I-set is simply given as part of the scene model.

4.4. Construction of partial interpretations

The schematic diagram of the processing involved in constructing partial interpretations is shown in Figure 4-3. The HLVS iterates the following steps in this stage:

(1) hypothesis generation,
(2) focus of attention,
(3) composite hypothesis construction,
(4) solution generation,
(5) action scheduling.

Detailed discussions of each step are presented in the following subsections.

4.4.1. Hypothesis generation

For each DE (hypothetical or verified) recorded in the iconic/symbolic database, the Iconic/Symbolic Database Manager (ISDM) evaluates all the rules that are "applicable".

Suppose I_0 is an instance of frame F. For each rule, say R_x, defined in frame F, the ISDM evaluates the <control-condition> part of rule R_x. If the evaluation result is true, the ISDM performs the following tasks:

(1) Compute the <hypothesis> part of rule R_x, and insert the computed hypothesis into the iconic/symbolic database.

(2) Insert the <action> part of rule R_x into the *Action List* which records all the actions waiting to be evaluated.

The actions in the action list are called *delayed actions*. For each delayed action, there is an associated hypothesis (computed at step 1) recorded in the iconic/symbolic database. Such a hypothesis is called the *cause of delay* of the action.

Note that for rules whose <hypothesis> part is nil, the <action> part is not put into the action list. Instead, the <action> is evaluated immediately. At the hypothesis generation stage, the ISDM evaluates, for each instance in the iconic/symbolic database, the <control condition> of every rule in the associated frame definition. (This strategy is not efficient. A more efficient strategy would evaluate only those <control condition> s whose values are affected by changes made to the attributes of the instance since the last time the <control condition>s were evaluated.)

The DE's in the iconic/symbolic database are combined into situations. All the situations are structured into the situation lattice. The Situation Lattice Database Manager (SLDM) updates the situation lattice whenever DE's are inserted into or removed from the iconic/symbolic database. The algorithm (3-1) for updating the situation lattice was presented in Section 3.4.

"Identical instances" may be created during the construction process of the HLVS. Two instances are *identical* if all the values filled in the slots of those instances are identical. It is necessary to detect identical instances and replace them by a single instance. This process is called *unification of instances,* and is performed during construction of composite hypotheses.

For example, a house group instance containing house instances H_0 and H_1 can be constructed from instance H_0 by first constructing a house group instance, say HG_0, which contains H_0 , and then expanding HG_0 to include house instance H_1 (see Figure 4-4(a)). An identical house group instance HG_1 can also be constructed from house instance H_1 (see Figure 4-4(b)).

One natural way to detect identical instances is to examine the P-set of a situation. For each situation selected by the focus of attention mechanism, the HLVS examines the instances in the P-set of the situation to find sets of identical instances.

The HLVS unifies identical instances as follows. All identical instances are first collected in a set, L. Then the HLVS selects one instance from the set L, say I_0. For each instance $I_x \in L$, the HLVS replaces every reference to I_x in the iconic/symbolic database by a reference to instance I_0.

Figure 4-5 illustrates the result of unifying HG_0 and HG_1 (assuming the HLVS chooses HG_0 as I_0).

4.4.2. Focus of attention

The focus of attention mechanism selects a situation with greatest strength from the situation lattice. If there are several situations with equal strength, the HLVS selects one arbitrarily.

For example, Figure 4-6 shows a situation lattice. There are two maximal consistent situations that can be selected (both situations have strength = 3). The HLVS can select either one (i.e., N_{10}, or N_{11}).

The situation selected by the focus of attention mechanism is given to the Composite Hypothesis Constructor to construct the composite hypothesis. The construction of composite hypotheses was discussed in Section 3.5.

4.4.3. Solution generation

The Solution Generator (SG) computes solutions for the composite hypothesis. The SG obtains/constructs instances to satisfy the composite hypothesis by one of the methods discussed in the following paragraphs.

First, the SG may discover an existing instance in the iconic/symbolic database that satisfies the composite hypothesis. In this case, the SG returns the instance found as the solution. In general, it may be necessary to search the iconic/symbolic database to find some instance which satisfies the composite hypothesis. However, since the composite hypothesis is constructed by taking the solution space of all the constraints imposed on the DE's participating in the situation (see Section 3.5), to find an existing instance which

satisfies the composite hypothesis, the SG needs only examine the P-set of the selected situation and use any instance in the P-set as the solution.

Suppose the SG cannot find any instance in the iconic/symbolic database that satisfies the composite hypothesis. There are two possibilities:

> (1) the target object of the composite hypothesis is a primitive object (such hypotheses are called *primitive hypotheses);*
>
> (2) the target object of the composite hypothesis is not a primitive object (such hypotheses are called *non-primitive hypotheses).*

In the first case, the SG first directs a top-down segmentation by providing to the LLVS the descriptions of the composite hypothesis. Then the SG creates instances based on the results of the LLVS. Finally, the instances created (if any) are returned as a solution.

In the second case, no top-down segmentation is performed. The SG simply returns the composite hypothesis as the solution.

4.4.4. Action scheduling

The Action Scheduler (AS) schedules the actions in the action list using the solution provided by the SG. Three possible types of solutions may be provided:

(1) nil,i.e., the hypothesis cannot be verified,
(2) an instance,
(3) a composite hypothesis.

In both the first and the second cases, the AS selects those <action>s in the action list whose "causes of delay" are in the P-set of the selected situation. Let the solution be I_0, the actions selected be $A_1, \ldots, A_n$, and their causes of delay be $H_1, \ldots, H_n$, respectively. The AS performs the selected actions sequentially:

(a) replace all the references to H_i in action A_i by I_0,

(b) evaluate A_i,

(c) remove H_i from the iconic/symbolic database, or update the attributes of H_i (we will discuss this in more detail in Section 4.5).

In the third case, the AS marks the composite hypothesis, say CH_0, as *partially processed* and inserts it into the iconic/symbolic database. The AS also marks the currently selected situation, say S_0, as *unconcluded.* The hypothesis CH_0 is said to be *derived from* the situation S_0. We will present a more detailed discussion of the effects of such processing in Section 4.4.4.1. Table 4-1 summarizes the terms defined in the previous paragraphs.

The removal of hypotheses from the iconic/symbolic database has the following side effects:

(1) If a hypothesis, say H_0, is removed from the database, then all the

Table 4-1. Glossary.

Primitive hypothesis:
A hypothesis whose target object is a primitive object.

Non-primitive hypothesis:
A hypothesis whose target object is a non-primitive object.

Unconcluded situation:
A situation which was selected by the focus of attention mechanism, but for which the Solution Generator cannot yet compute a solution.

Partially processed hypothesis:
A composite hypothesis, recorded in the iconic/symbolic database, which is computed for some unconcluded situation.

situations in the situation lattice whose P-sets contain H_0 are also removed from the situation lattice.

(2) If an unconcluded situation is removed from the situation lattice in (1), then the hypotheses which were derived from the situation are also removed from the iconic/symbolic database.

The updating of attributes of hypotheses is implemented by removing the original hypothesis and inserting a new hypothesis.

When all the actions selected are evaluated, the action scheduler terminates, and the next cycle of hypothesis construction begins.

4.4.4.1. Computing solutions for a non-primitive composite hypothesis

The SG does not directly propose solutions for a non-primitive composite hypothesis. Instead, a top-down parsing approach is used to compute the solution. Suppose the composite hypothesis constructed for a situation , say S_0, is CH_a. To compute the solution for CH_a, we first generate a set of hypotheses $H_i, 1 \leq i \leq n$ and compute the solution for each H_i. The solution for CH_a can be computed from the solutions for $H_i, 1 \leq i \leq n$.

To support such an approach, we associate with each non-primitive frame a *decomposition strategy* (represented as a rule) which describes how to generate a new set of hypotheses to be verified, and how to compute a solution for the non-primitive frame using the solutions for the generated hypotheses.

For example, the rule for the decomposition strategy of a RECTANGULAR-HOUSE frame is

Rule $R_{first\text{-}order\text{-}properties}$:
<control-condition> : true,
<hypothesis> :
  $H = F_0(RECTANGLE, self)$,
<action> :
  if H=nil then conclude(nil)
  else conclude(make-instance(RECTANGULAR-HOUSE,H)).

This rule indicates that a RECTANGULAR-HOUSE instance can be created

if a RECTANGLE instance which satisfies the attributes specified by F_0 is created.

As discussed in Section 4.4.4, the Action Scheduler (AS) marks the non-primitive composite hypothesis as partially processed and inserts it into the iconic/symbolic database. The AS also marks the situation selected as unconcluded. Partially processed hypotheses and unconcluded situations are processed by other modules of the HLVS in the following ways:

(1) If a situation, say S, is marked as "unconcluded", then all the situations in the situation lattice which are less than S are also marked as unconcluded. The focus of attention mechanism does not select any unconcluded situation. This strategy is based on the observation that if no conclusion can be drawn from the analysis of a situation, say S, then the analysis of all the situations which are "less than" S (i.e., composed of a subset of the instances of S) can be postponed.

For example, by marking situation N_{10} in Figure 4-6 as unconcluded, all the situations that are less than N_{10} are also marked as unconcluded (i.e., $N_i, H_i, 1 \leq i \leq 3$).

(2) The function "conclude" indicates that a solution, say I_{sol}, has been computed for an unconcluded situation, say S. Whenever this function is evaluated, the HLVS schedules S as the situation to be selected in the next iteration cycle and the solution proposed for the composite hypothesis of this situation is I_{sol}.

(3) Since a partially processed hypothesis, say H, is the composite hypothesis constructed for some unconcluded situation, S, H should not participate in the formation of new situations with any DE's in the P-set of S. HLVS uses the more efficient strategy of not allowing a partially processed hypotheses to participate in the formation of *any* situations.

(4) In the hypothesis generation process, only the rules which describe the decomposition strategy can be evaluated for partially processed hypotheses.

All the hypotheses generated are inserted into the iconic/symbolic database.

(5) The removal of a partially processed hypothesis from the iconic/symbolic database causes the removal of all the hypotheses in the database which are generated by the decomposition strategy.

Suppose, for example, that the situation N_{10} shown in Figure 4-6 is selected by the focus of attention mechanism and the composite hypothesis constructed, say CH_a, is:

target object : RECTANGULAR-HOUSE;
...

Since RECTANGULAR-HOUSE is not a primitive frame, the SG returns CH_a as the solution to the situation N_{10}. The AS marks N_{10} as unconcluded and inserts CH_a into the iconic/symbolic database.

At the subsequent hypothesis generation process, CH_a activates the rule $R_{first\text{-}order\text{-}properties}$ in the RECTANGULAR-HOUSE frame and creates hypothesis H_9:

target object : RECTANGLE;
...

Figure 4-7 shows the relation between CH_a and H_9 and the action which is delayed by H_9. The resulting situation lattice is shown in Figure 4-8.

Suppose a RECTANGLE instance, say I_R, is proposed to H_9 by the SG. The AS evaluates the action whose cause of delay is H_9 and:

(1) creates a RECTANGULAR-HOUSE instance, say I_{RH},

(2) evaluates the function "conclude".

The evaluation of the function "conclude" indicates to the HLVS that situation N_{10} is to be scheduled in the next iteration cycle and the solution proposed for CH_a is I_{RH}.

At the next iteration, the SG proposes I_{RH} to the hypotheses in the P-set of N_{10} (i.e., H_1, H_2, H_3). Those actions whose causes of delay are H_1, H_2, and H_3 are now evaluated by the Action Scheduler. Suppose H_1, H_2, and H_3 are removed after the evaluation of these actions. Figure 4-9 shows the resulting situation lattice. Note that this is usually the case when an appropriate solution is proposed to the hypotheses.

The processing of partially processed hypotheses and unconcluded situations are summarized in Table 4-2.

4.5. A taxonomy of actions

In this section, we discuss a taxonomy of the actions that are often used to specify the scene domain knowledge. The term action in this section refers to the activities described in the <action> part of a rule.

One type of action is the filling in of attributes of an instance. For example, a rule in the HOUSE-GROUP frame is:

Table 4-2. Summary.

Unconcluded situation:

- Will not be selected by the focus of attention mechanism.
- If a solution is proposed by the SG for some unconcluded situation, the HLVS schedules that situation in the next iteration cycle.

Partially processed hypothesis:

- A composite hypothesis for some unconcluded situation.
- Recorded in the iconic/symbolic database.
- Does not participate in the formation of any situations.
- Removal of a partially processed hypothesis, *H*, causes the removal of all the hypotheses generated by *H*.

<control-condition> : true
<hypothesis> : $H \doteq$ AR(self,ROAD),
<action> : self.along-road = *H*.

This rule specifies that if a ROAD instance which satisfies *H* is found, fill it in the slot "along-road" of the HOUSE-GROUP instance.

In addition to filling in attributes, actions often create new instances or unify several instances (as described in Section 4.4.1). Such actions are described by two functions:

- "make-instance" : create an instance and insert it into the iconic/symbolic database;

- "unify-instance" : unify a list of instances in the iconic/symbolic database into a single instance.

For example, a rule in the RECTANGLE frame is:

<control-condition> : IS-RECT-HOUSE(self)
<hypothesis> : nil,
<action> :
make-instance(RECTANGULAR-HOUSE,F(self)).

This rule describes the following piece of knowledge:

"If a RECTANGLE instance which satisfies the IS-RECT-HOUSE criteria is created, then create a RECTANGULAR-HOUSE instance using function F and insert it into the iconic/symbolic database."

Similarly, the following piece of knowledge:

"If more than one HOUSE-GROUP instance is filled in the "belongs-to" slot of a HOUSE instance, replace it by another HOUSE-GROUP instance which is created by the function COMBINE-H."

can be described by the following rule in the HOUSE frame:

<control-condition> :
if number-of-elements(self.belongs-to) > 1,
<hypothesis> : nil,
<action> :
unify-instance(self.belongs-to,COMBINE-H(self.belongs-to)).

Another class of actions deals with the removal of hypotheses and the updating of the attributes of hypotheses. Usually, hypotheses are removed by the Action Scheduler after the Solution Generator proposes solutions to them. However, instead of always removing hypotheses when no acceptable solution is found, we may want to update the attributes of the original hypotheses when more information is available. The function "update" is used to describe the updating of the attributes of a hypothesis.

For example, consider the following rule:

<control-condition> : ...
<hypothesis> : H = F(self)
<action> :
    if H = nil then update(H,CS_1)
    else ...

The action specifies that if the solution proposed for H is nil, then the AS replaces some attributes of hypothesis H by CS_1. However, H is not removed from the iconic/symbolic database. The <action> part is inserted again into the action list (its cause of delay is H.)

There is yet another category of actions which specifies the constraints on the evaluation of multiple rules. We describe this type by an example.

Any instance of a HOUSE-GROUP frame can be "along" at most one ROAD instance. Given a HOUSE-GROUP instance, say I_{HG}, we may not yet know the location of the road along I_{HG} ,i.e., at location F_l or at location F_r (see Figure 4-10). One strategy is to create hypotheses about a ROAD at

both locations. However, once one hypothesis is verified, the other hypothesis must be removed.

The above knowledge is represented as follows:

Rule R_1.
<control-condition> : true,
<hypothesis> : $H_1 = F_l$(self),
<action> : self.along-road = H_1,

Rule R_2.
<control-condition> : true
<hypothesis> : $H_2 = F_r$(self),
<action> : self.along-road = H_2.

In addition, the following rule for the HOUSE-GROUP frame constrains the simultaneous evaluation of R_1,R_2:

Rule $R_{control}$.
<control-condition> :
not-null(anyone(R_1,R_2)),
<hypothesis> : nil,
<action> :
remove-all(anyone(R_1,R_2)).

where anyone(R_1,R_2)=
if is-evaluated(R_1) then R_2
else if is-evaluated(R_2) then R_1
else nil

The above rule specifies that whenever one of the <action> parts of the rules R_1 or R_2 is evaluated, rule $R_{control}$ is evaluated which causes the removal of all the hypotheses that are created by the evaluation of R_1.<hypothesis>

or R_2.<hypothesis>.

Suppose a HOUSE-GROUP instance is created. The instance activates rules R_1 and R_2 and generates two hypotheses about the ROAD object. Whenever the SG proposes a ROAD instance to one of the hypotheses, the AS evaluates one of the delayed actions, and causes the removal of the other hypothesis.

We summarize the actions discussed in this section in Table 4-3.

4.6. Pursuing alternative hypotheses

It is possible that several hypotheses may be generated at the same time. This can be represented as the following rule:

Table 4-3. A taxonomy of actions

Action Type	Example
Attributes	Filling in of attributes in an instance.
Instances	Create instances. Unify instances.
Hypotheses	Remove hypotheses. Update hypotheses.
Rules	Constrain the simultaneous evaluation of several rules.

```
if <control-condition> then
        <hypothesis 1> <action 1>
or
        <hypothesis 2> <action 2>
or
        <hypothesis n> <action n>
```

Whenever <control-condition> evaluates to true, all of the <hypothesis>s can be generated. These hypotheses are called *alternative hypotheses* and we assume that at most one of the hypotheses is in fact true. However, it is difficult to decide which one should be pursued first, since a promising selection may turned out to be incorrect as new facts (generated by resegmentation) are obtained.

In SIGMA, all the alternative hypotheses are generated and participate in the hypothesis integration process. However, the associated actions of these alternative hypotheses are not evaluated (put in the delayed-action queue). When any one of the alternative hypotheses is verified, it is left to the associated action to decide whether other alternative hypotheses should be pruned. On the one hand, this strategy allows multiple alternative hypotheses to be pursued simultaneously. On the other hand, expert domain knowledge, which can be described in a rule, can be used to prune unpromising hypotheses when enough facts are known.

4.7. The selection of good interpretations

Potentially, SIGMA could construct all possible interpretations for the image. It is natural to require that no region be interpreted as two different objects in the scene model. However, in SIGMA, a region may be interpreted as several objects (e.g., an elongated region might be interpreted both as a road or a driveway). Intersecting image structures may be used to construct DE's whose iconic descriptions should never intersect. A pair of DE's whose iconic descriptions intersect while the scene model specifies otherwise are called *conflicting* DE's. The associated interpretations are called *alternative* interpretations.

For a set of conflicting DE's, we need to select a DE which "best" interprets the image. It is possible to design an algorithm to select such "best" interpretations. However, we did not investigate this issue in SIGMA. Instead, we model the final selection process as a database query answering process. A program (QAM) was developed to answer simple queries about DE's in the interpretation network and to display the iconic descriptions of the DE's selected.

5. Examples

5.1. Introduction

This section presents detailed examples of the application of SIGMA to the analysis of a high resolution aerial image to locate houses, roads, and driveways in a suburban scene.

We first present an example of the initial segmentation process. Then we discuss how the HLVS analyzes the image in several typical situations. Finally, we show the results of analysis by SIGMA on an aerial image.

5.2. Initial segmentation

The image used in the example is a 250 * 140 window of an aerial image (Figure 5-1) with intensities in the range of 0 to 63. The scene contains houses, roads, and driveways.

5.2.1. Initial segmentation goals

We want to locate houses and roads in the image. Since their appearances are either compact rectangles or elongated rectangles, and they are usually brighter than the background, the following hypotheses are used as the I-set of the initial segmentation process:

/* extract compact and bright rectangles */
hypothesis H_{blob}:
target object = rectangle,
in-window = whole image,
rectangle.elongatedness $\leq$ 10,
rectangle.compactness $<$ 18,
rectangle.region-contrast $>$ 3,
180 $<$ rectangle.area-of $<$ 360.

/* extract bright and elongated rectangles */
hypothesis H_{ribbon}:
target object = rectangle,
in-window = whole image,
8 $<$ rectangle.width $<$ 20
rectangle.elongatedness $>$ 10,
rectangle.length $>$ 10,
rectangle.compactness $\geq$ 18,
rectangle.region-contrast $>$ 3.

5.2.2. Verifying hypothesis H_{blob}

The Initial Segmentation Controller (ISC) first selects hypothesis H_{blob}. The ISC activates the LLVS to compute image primitives that satisfy hypothesis H_{blob}. The LLVS selects the following segmentation operators arranged in descending order of their priorities as follows:

Blob finder
Upper threshold method

The Ribbon finder and the Lower threshold method are not selected since their selection criteria evaluate to false.

The LLVS activates the Blob finder first. The Blob finder first convolves the original image with a Laplacian operator. Then it computes the positive connected regions in the convolved image (Figure 5-2). The regions computed by the Blob finder which satisfy the constraints of H_{blob} are shown in Figure 5-3.

Since the set of results computed by the Blob finder is not empty, the LLVS returns the computed regions to the HLVS. The Upper threshold method is not evaluated.

5.2.3. Verifying hypothesis H_{ribbon}

The ISC then selects hypothesis H_{ribbon}. The ISC activates the LLVS to compute regions which satisfy hypothesis H_{ribbon}. The segmentation operators selected by the LLVS for this task arranged in descending order of their priorities are as follows:

Ribbon finder
Upper threshold method

The Blob finder and the Lower threshold method are not selected since their selection criteria evaluate to false.

The LLVS activates the Ribbon finder first. The Ribbon finder first computes the skeletons of the positive regions shown in Figure 5-2. The resulting skeletons are shown in Figure 5-4.

Finally, the Ribbon finder decomposes these skeletons and computes the skeletons for the ribbons. Figure 5-5 shows the skeletons of the ribbons computed by the Ribbon finder which satisfy the constraints of hypothesis H_{ribbon}.

Since the set of results computed by the Ribbon finder is not empty, the LLVS returns the computed regions to the HLVS. The Upper threshold method is not evaluated.

5.2.4. Generating instances

The ISC collects the results computed by the LLVS, creates RECTANGLE instances, and inserts them into the iconic/symbolic database.

There are 26 RECTANGLE instances created at this stage. Figure 5-6 shows the iconic descriptions of these instances. Note that some of the instances intersect.

5.3. Constructing partial interpretations

A situation is classified into one of the following classes based on how the Solution Generator computes its proposed solution:

Case 1: The SG discovers an existing instance in the iconic/symbolic database which satisfies the given composite hypothesis.

Case 2: The SG cannot find any instance in the iconic/symbolic database which satisfies the given composite hypotheses. The composite hypothesis is non-primitive.

Case 3: The SG cannot find any instance in the iconic/symbolic database which satisfies the given composite hypothesis. The composite hypothesis is primitive.

Case 4: The SG obtains the solution from the previous iteration (i.e., the solution for an unconcluded situation is now computed.)

5.3.1. Case 1--Discovering an existing instance

Consider the situation shown in Figure 5-7. The relations between the DE's shown in this figure are described in Table 5-1.

Figure 5-8 shows the portion of the interpretation-network which is related to this situation.

Assume the focus of attention mechanism selects situation S_1 whose P-set is as follows:

$$\left\{DE_1, DE_3, DE_7\right\}.$$

Suppose the composite hypothesis, say CH_a, computed for S_1 is :

target object = ROAD,
...

Since the P-set of the situation S_1 contains an instance, DE_7, the SG proposes it as the solution to the composite hypothesis constructed for this situation.

The AS activates those actions whose causes of del are DE_1 and DE_3 respectively. Figure 5-9 shows the resulting interpretation network. Note that hypotheses DE_2 and DE_4 are removed. This is caused by a control rule in the HOUSE-GROUP frame which specifies that each HOUSE-GROUP instance can be along at most one road.

5.3.2. Case 2--Decomposing a hypothesis

Consider the situation shown in Figure 5-10. The relations between the DE's shown in this figure are described in Table 5-2.

Figure 5-11 shows a portion of the interpretation network related to this situation.

Assume the focus of attention mechanism selects the situation S_1 whose P-set is

$$\left\{DE_1, DE_2\right\}.$$

Assume the composite hypothesis, say CH_a, computed for S_1 is

target object = DRIVEWAY,
...

The SG cannot find any existing instance that satisfies CH_a. Since CH_a is non-primitive, the AS marks it as partially processed and inserts it into the

iconic/symbolic database.

At the subsequent iterations, CH_a activates the rule $R_{first-order-properties}$ of frame DRIVEWAY to generate hypothesis DE_3:

databaseentity DE_3:
target object : RECTANGLE,
...
end–database–entity.

Suppose the action which is delayed by DE_3 is $A_{first-order-properties}$. We will revisit this example in Section 5.3.4. Note that DE_3 can participate in the formation of situations with other DE's in the iconic/symbolic database. Figure 5-12 shows the resulting interpretation network after DE_3 and CH_a are inserted into the iconic/symbolic database. Note that CH_a is marked as partially processed hypothesis. Table 5-3 summarizes the relations between the DE's, action $A_{first-order-properties}$, and S_1.

5.3.3. Case 3--Directing the segmentation

Suppose the composite hypothesis, say CH_a, given to the SG is primitive. The SG activates the LLVS to compute regions which satisfy the constraints provided by the SG. The regions computed by the LLVS are used by the SG to create RECTANGLE instances. The SG then proposes those created instances which satisfy the constraints of CH_a as solutions. If no instance is computed, the SG proposes nil as the solution. We illustrate the process used

by our system in the following two examples.

Suppose the composite hypothesis, say CH_a, given to the SG is:

target object = RECTANGLE,
in window : W_1,
rectangle.elongatedness $\leq$ 10,
rectangle.compactness $<$ 18,
275 $<$ rectangle.area-of $<$ 325.

The window W_1 is shown in Figure 5-13.

The LLVS first activates the Blob finder and fails to compute any region. Then the LLVS activates the Upper threshold method to compute regions. A region is successfully computed by setting the threshold value at 24. Figure 5-14 shows some of the intermediate results of the segmentation process. The measurements (the area and the compactness of a region) are shown for the largest region extracted at each specified threshold value.

The LLVS returns the computed region to the SG. The SG checks the features of the region and creates a RECTANGLE instance DE_{RECT} and propose it as the solution. Figure 5-15 shows the RECTANGLE instance created by the SG.

Suppose the composite hypothesis CH_a is again given to the SG. However, the window W_1 is as shown in Figure 5-16.

The LLVS activates the Blob finder, the Upper threshold method, and the Lower threshold method and cannot compute any region which satisfies

the given constraints. The LLVS returns "nil" to the SG. The SG then proposes nil as the solution.

5.3.4. Case 4--Analyzing an unconcluded situation

Consider the interpretation network shown in Figure 5-12. Suppose that at some other iteration the SG computes a solution, say I_0, for DE_3. Action $A_{first-order-properties}$ is now evaluated by the AS.

Two possible outcomes can be produced by the evaluation of $A_{first-order-properties}$. First, the evaluation of action $A_{first-order-properties}$ generates a solution, say I_1, for CH_a. This causes the HLVS to analyze the unconcluded situation S_1 in the next iteration. The SG will propose I_1 as the solution to CH_a, the composite hypothesis of S_1.

Figure 5-17 shows the resulting interpretation network in this case. The unconcluded situation S_1, the partially processed hypothesis CH_a, and the hypothesis DE_3 generated by the "decomposition method" are all removed.

Second, suppose no solution is generated by the evaluation of $A_{first-order-properties}$. Instead, the evaluation cause changes to be made to the attributes of DE_3. In this case, situation S_1 is removed from the situation lattice and new situations are constructed. Suppose DE_{3a} is the updated hypothesis. Figure 5-18 shows the resulting interpretation network in this case.

5.4. A complete example

In this section, we present the result of applying our image interpretation program to the image shown in Figure 5-1. No explicit goal is given to the system. The analysis terminates when all the hypotheses created are verified or refuted.

Figure 5-6 shows the RECTANGLE instances generated by the initial segmentation process. Figure 5-19 shows those RECTANGLE instances which are interpreted as RECTANGULAR-HOUSE instances (requiring that 200<rectangle.area-of<400) , and Figure 5-20 shows those RECTANGLE instance which are interpreted as VISIBLE-ROAD-PIECE instances (requiring that 6<rectangle.width<12). No RECTANGLE instances are interpreted as DRIVEWAY instances.

Instead of showing the processing of each situation by the program, we show only the processing of several interesting situations.

In the scene model, two HOUSE-GROUP instances are identical if they both share a common HOUSE instance and should be unified to a single instance. Figure 5-21(a) shows such an example. Let P_1 and P_2 denote two HOUSE instances, R_1 and R_2 two HOUSE-GROUP instances, and DE_i a HOUSE hypothesis.

Each HOUSE-GROUP instance creates hypotheses about more houses that belong to it. The process to unify the house groups is as follows:

(1) The situation whose P-set is

$$\{DE_1, P_2\}$$

is selected by the focus of attention mechanism.

(2) SG proposes HOUSE instance P_2 as the solution to the composite hypothesis of situation S_1. The evaluation of the action which is delayed by DE_1 fills P_2 in the "contains" slot of HOUSE-GROUP instance R_1.

(3) Since P_1 "belongs to" two HOUSE-GROUP instances at the subsequent iteration, the evaluation of a rule in HOUSE frame unifies R_1 and R_2.

Let us denote the resulting HOUSE-GROUP instance by R_1. Figure 5-22 shows the result of the analysis.

Figure 5-23 shows another example. Resegmentation of the image is required in this example. Let R_i denote a HOUSE-GROUP instance, P_i a HOUSE instance, DE_i a HOUSE hypothesis. Also let CH_i denote a partially processed hypothesis, and T_1 a RECTANGLE instance. These DE's are not shown in Figure 5-23. They are used later in this example.

The processes to activates the LLVS to process the image are as follows:

(1) Situation S_1 whose P-set is

$$\{DE_1, DE_3\}$$

is selected. Since the composite hypothesis (target object is HOUSE object) is non-primitive, a partially processed hypothesis, say CH_1, is generated.

(2) At the next iteration, the evaluation of the rule $R_{specialization-strategy}$ of the HOUSE frame generates a hypothesis DE_5 whose target object is RECTANGULAR-HOUSE (Figure 5-24(a)).

(3) Situation S_2 whose P-set contains DE_5 is selected. Again, a partially-processed hypothesis, say CH_2, about RECTANGULAR-HOUSE is generated.

(4) At the following iteration, the evaluation of the rule $R_{first-order-properties}$ of RECTANGULAR-HOUSE frame generates a hypothesis DE_6 whose target object is RECTANGLE (Figure 5-24(b)).

(5) The SG activates the LLVS to segment the image. A region is computed by the LLVS (see Figures 5-13, 14, 15). The SG creates a RECTANGLE instance T_1.

(6) The evaluation of the <action> of $R_{first-order-properties}$ creates a RECTANGULAR-HOUSE instance P_4. Since a solution is now ready for the unconcluded situation S_2, the HLVS schedules it to be processed next. Afterwards, since a solution is now ready for the unconcluded situation S_1, the HLVS schedules it to be processed next. Now, the actions delayed by DE_1 and DE_3 can be evaluated. The resulting interpretation network is shown in Figure 5-24(c).

(7) P_4 "belongs to" two HOUSE-GROUP instances. At the subsequent iteration, the evaluation of a rule in the HOUSE frame unifies R_1 and R_2.

Figure 5-25 shows the resulting HOUSE-GROUP instance.

In the scene model, every ROAD instance is smoothly extended from one ROAD-TERMINATOR instance to another ROAD-TERMINATOR instance. A ROAD-TERMINATOR is defined to be the boundary of the image. We present an example in the following paragraphs.

The extension of ROAD instances is similar to the merging of two HOUSE-GROUP instances discussed above. Figure 5-26 shows two ROAD instances R_1 and R_2. P_1 and P_2 are two ROAD-PIECE instances. DE_i denotes a ROAD-PIECE hypothesis. The extending of ROAD instance R_1 activates the merging of R_1 and R_2 into one ROAD instance (Figure 5-27).

Figure 5-28 shows another case. R_1 and R_2 are two ROAD instances. DE_1 is a ROAD-PIECE hypothesis generated by R_1. Since R_2 is not "connected" to R_1, hypothesis DE_1 is modified as shown in Figure 5-29.

Figure 5-30 shows yet another case. Road instance R_1 cannot be extended any longer. When this is detected, the original ROAD-PIECE hypothesis is removed and a ROAD-TERMINATOR hypothesis is generated.

Figure 5-31 shows another example. Let DE_r denote a ROAD instance, DE_h a HOUSE instance, DE_{re} a RECTANGLE instance, and DE_i a DRIVEWAY hypothesis. House instance DE_h and ROAD instance DE_r create hypotheses DE_1 and DE_2 about the DRIVEWAY object respectively. There is no DRIVEWAY instance in the iconic/symbolic database which satisfies these hypotheses. However, there is a RECTANGLE instance, DE_{re}, which, if interpreted as a DRIVEWAY object, would satisfy these hypotheses. Note that DE_{re} is not interpreted as a DRIVEWAY object, a VISIBLE-ROAD-PIECE, or a RECTANGULAR-HOUSE since there are not enough distinguishing features of DE_{re} to make these interpretations.

The HLVS performs the analysis as follows:

(1) A composite hypothesis, CH_1, is first constructed for the situation whose P-set is

$$\left\{DE_1, DE_2\right\}.$$

(2) A hypothesis, DE_3, about the RECTANGLE object is created by the composite hypothesis CH_1.

(3) DE_{re} satisfies DE_3. A DRIVEWAY instance DE_{dr} is created by the <action> part of the rule $R_{first-order-properties}$ of the DRIVEWAY frame. The DRIVEWAY instance DE_{dr} satisfies both DE_1 and DE_2. Figure 5-32 shows the resulting interpretation network after DE_1 and DE_2 are removed.

The resulting interpretation network is shown in Figure 5-33. The iconic descriptions of the instances created during the analysis are shown in Figures 5-34 and 5-35.

Finally, we present two examples of the final selection stage of the program. Figure 5-36(a) shows a ROAD instance whose length is longer than 100. Instances of related objects are shown in Figure 5-36(b),(c), and(d).

Figure 5-37(a) shows a HOUSE-GROUP instance with more than four houses. Instances of related objects are shown in Figure 5-37(b) and (c).

6. Conclusions

This paper has described a model for the development of image understanding systems that involves representing scene domain knowledge using frames and controlling the actions of the system by hypothesis integration. Using such a framework, we developed a flexible image understanding system called SIGMA which performs both top-down(goal-oriented) image analysis and bottom-up construction of composite image structures, and demonstrated the system's performance on an aerial image of a suburban scene.

Developing computer systems for visual applications is one way to investigate how humans see, and also to make computers more useful. As pointed out by many researchers [Hall79], [Binf82], image analysis systems usually consist of several types of modules: low level vision modules(e.g., segmentation) and high level vision modules(e.g., matching, inference). This research leads to the conclusion that a powerful vision system should rely on a balance of performance between these two types of modules. The low level modules should provide descriptive information about the image to the high level modules and the high level modules should provide "hints" about image structures to the low level modules. This research is only a small step toward the construction of general vision systems.

REFERENCES

[Ball82]

D. H. Ballard and C. M. Brown, *Computer Vision,* Prentice-Hall, Inc., Englewood Cliffs, N.J., 1982.

[Barr76]

H. G. Barrow and J. M. Tenenbaum, *MSYS:A system for reasoning about scenes,* Tech. Note 121, SRI International, Menlo Park, CA, April 1976.

[Barr81]

H. G. Barrow and J. M. Tenenbaum, Computational vision, *Proc. of the IEEE* **69**, 5, pp. 572-595, May 1981.

[Binf82]

T. O. Binford, Survey of model-based image analysis systems, *The International Journal of Robotics Research* **1**, 1, pp. 18-64, Spring 1982.

[Boll76]

R. C. Bolles, *Verification vision within a programmable assembly system,* Memo AIM-295, Stanford Artificial Intelligence Laboratory, Dec. 1976.

[Davi77]

R. Davis and J. King, An overview of production systems, in *Machine Intelligence 8:Machine representations of knowledge,* ed. Elcock and Michie, Wiley, NY., 1977.

[Garv76]

T. D. Garvey, *Perceptual strategies for purposive vision,* Tech. Note 117, SRI International, Menlo Park, CA, Sept. 1976.

[Gold83]

A. Goldberg and D. Robson, *Smalltalk-80, the Language and its Implementation,* Addison Wesley, Reading, MA., 1983.

[Hall79]

E. L. Hall, *Computer Image Processing and Recognition,* Academic Press, New York, 1979.

[Hans78]

A. R. Hanson and E. M. Riseman, VISIONS: A computer system for interpreting scenes, pp. 303-333 in *Computer Vision Systems,* ed. A. Hanson and E. Riseman, Academic Press, New York, 1978.

[Hend79]

G. G. Hendrix, Encoding knowledge in partitioned networks, pp. 51-92 in *Associative Networks: Representation and Use of Knowledge by Computers,* ed. N. V. Findler, Academic Press, New York, 1979.

[Hwan84]

S. S. Vincent Hwang, Evidence accumulation for spatial reasoning in aerial image understanding, Ph. D. thesis, University of Maryland, College Park, Dec. 1984.

[McKe84]

D. M. McKeown, Jr., W. A. Harvey, and J. McDermott, *Rule based interpretation of aerial imagery,* Proc. of IEEE Workshop on Principles of Knowledge-Based Systems, Denver, Colorado, Dec. 1984.

[Mins75]

M. Minsky, A framework for representing knowledge, pp. 211-277 in *The Psychology of Computer Vision,* ed. P. H. Winston, McGraw Hill, New York, 1975.

[Moor79]

R. C. Moore, *Reasoning about knowledge and action,* Tech. Note 191, AI Center, SRI International, Menlo Park, CA, 1979.

[Naga80]

M. Nagao and T. Matsuyama, *A Structural Analysis of Complex Aerial Photographs,* Plenum Press, New York, 1980.

[Nazi84]

A. M. Nazif and M. D. Levine, Low level image segmentation:an expert system, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **6**, pp. 555-577, Sept. 1984.

[Nils80]

N. J. Nilsson, *Principles of Artificial Intelligence,* Tioga, Palo Alto, CA, 1980.

[Self82]

P. G. Selfridge, Reasoning about success and failure in aerial image understanding, Ph. D. thesis, University of Rochester, 1982.

[Smit78]

R. G. Smith and R. Davis, *Distributed problem solving: The contract net approach,* Report No. STAN-CS-78-667, Computer Science Department, Stanford University, Sept., 1978.

[Stee79]

L. Steels, Reasoning modeled as a society of communicating experts, Ph. D. thesis, TR-542, Artificial Intelligence Laboratory, MIT, June, 1979.

[Wein80]

D. Weinreb and D. Moon, *Flavors: Message passing in the LISP machine,* Memo. 602, MIT Artificial Intelligence Laboratory, MIT, Nov. 1980.

[Wino75]

T. Winograd, Frame representations and the declarative/procedural controversy, pp. 185-210 in *Representation and Understanding,* ed. D. G. Bobrow and A. M. Collins, Academic Press, New York, 1975.

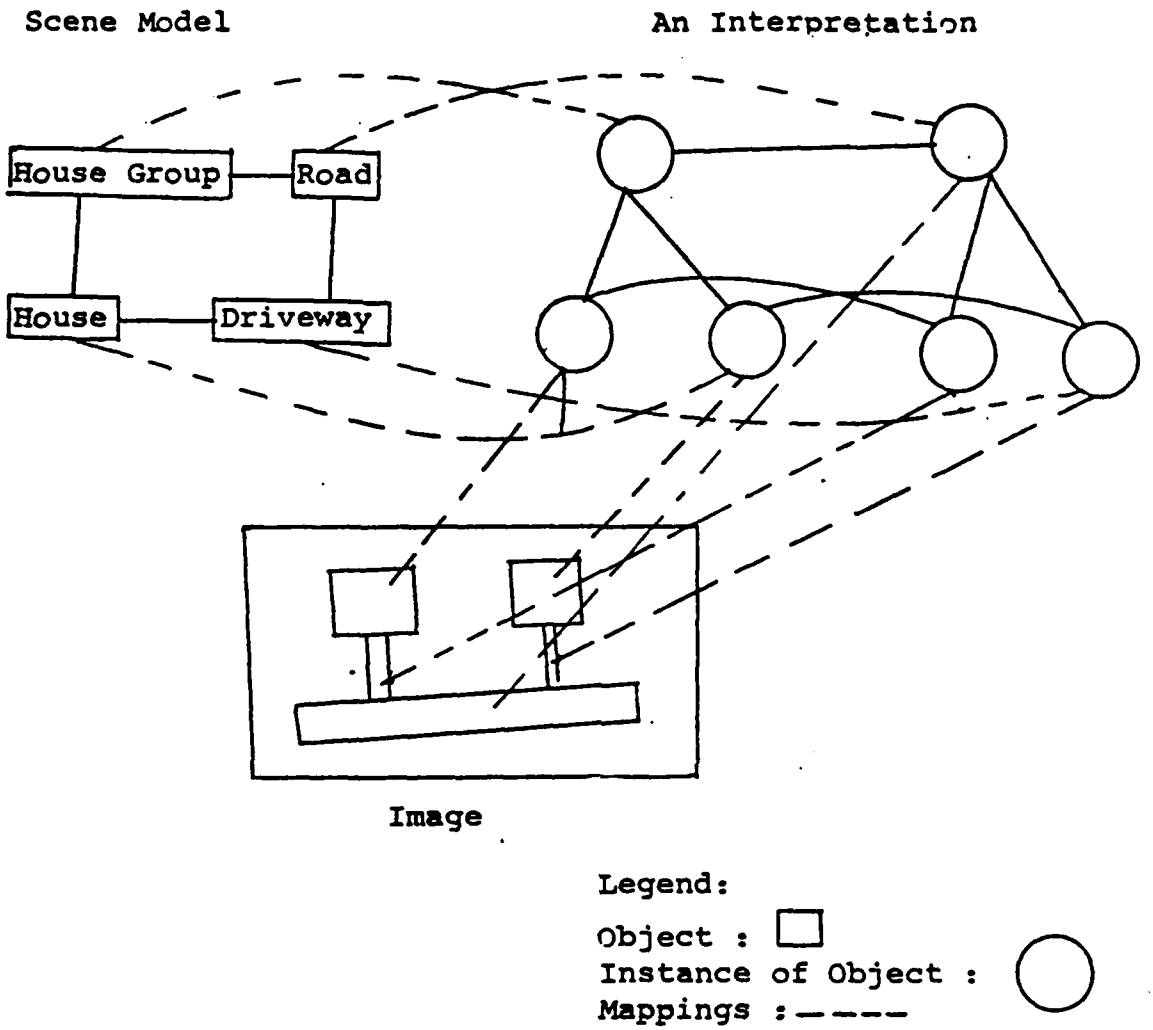


Figure 1-1. Mappings between the scene and the image.

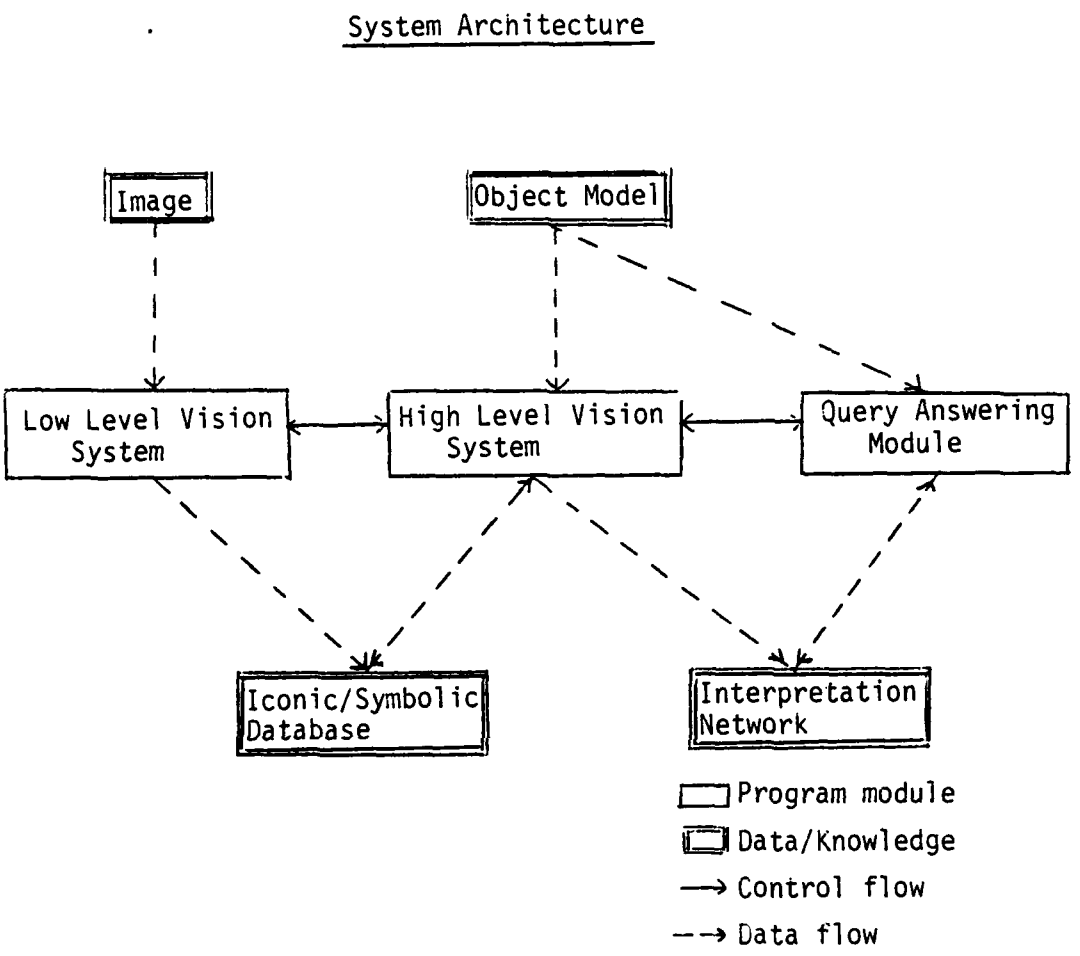


Figure 1-2. System architecture for the SIGMA image understanding system.

frame RECTANGULAR-HOUSE;
    rules:
        $F_{rectangle}$;
    links:
        AKO : HOUSE;
end-frame

frame L-SHAPED-HOUSE;
    rules:
        $F_{L-shape}$;
    links:
        AKO : HOUSE;
end-frame

frame HOUSE;
    slots:
        centroid;
        shape-description;
        front-of-house;
        connecting-driveway;
    rules:
        $F_{driveway}$;
    links:
        CAN-BE : RECTANGULAR-HOUSE, L-SHAPED-HOUSE;
end-frame

Figure 2-1 Frame definitions for HOUSE, RECTANGULAR-HOUSE, and L-SHAPED-HOUSE.

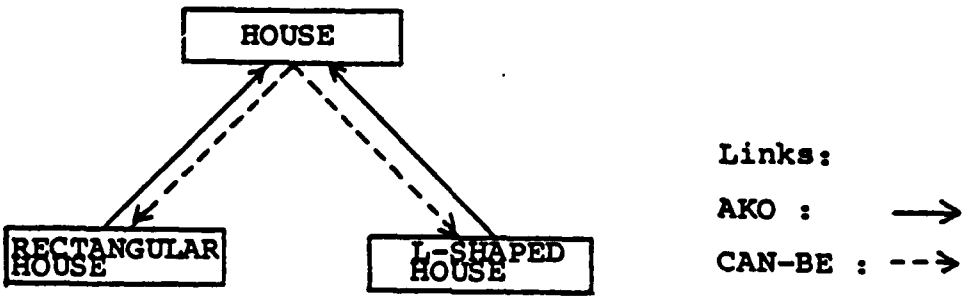


Figure 2-2 Links between HOUSE, RECTANGULAR-HOUSE and L-SHAPED-HOUSE frames.

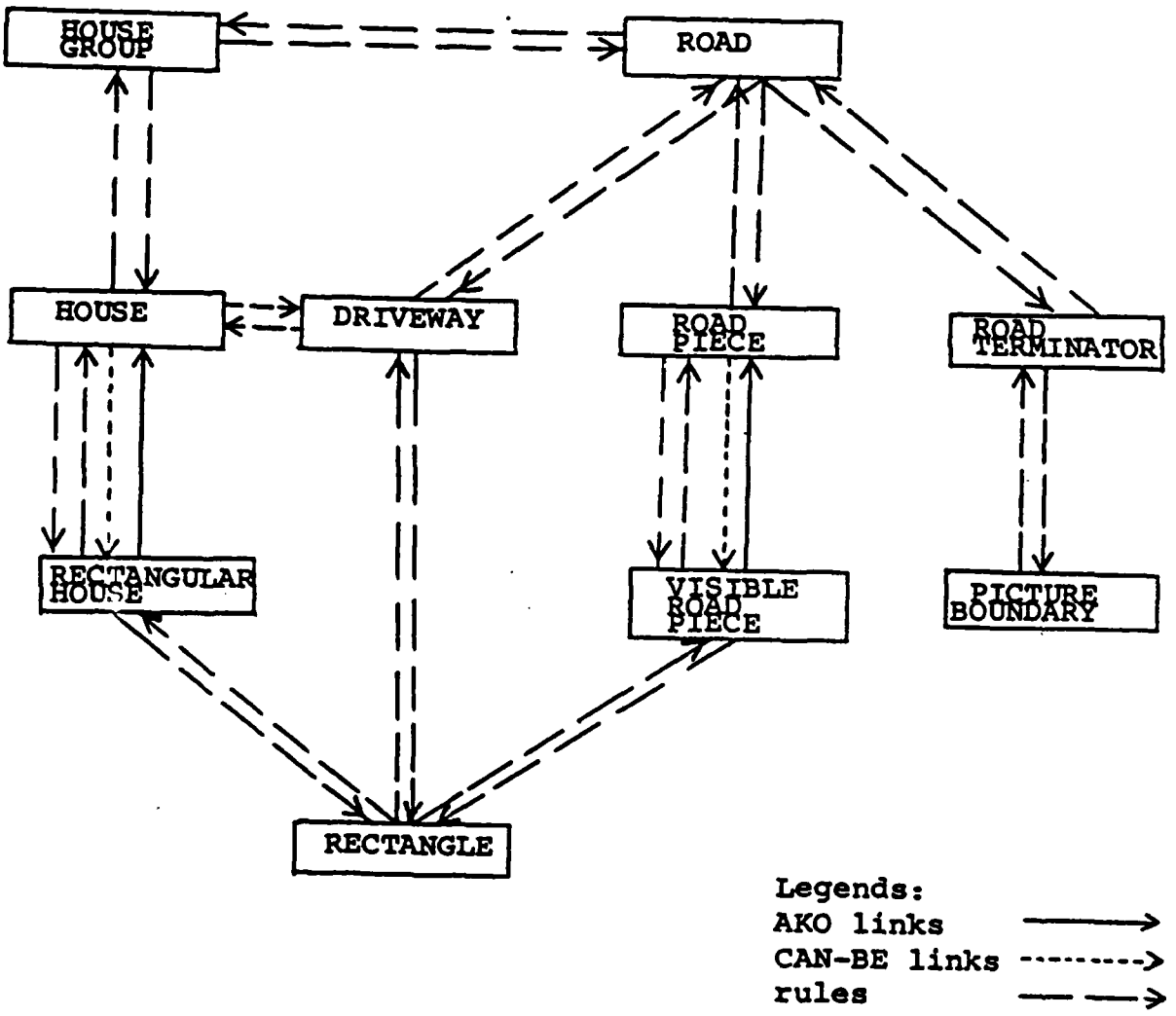


Figure 2-3 A model of a suburban housing development.

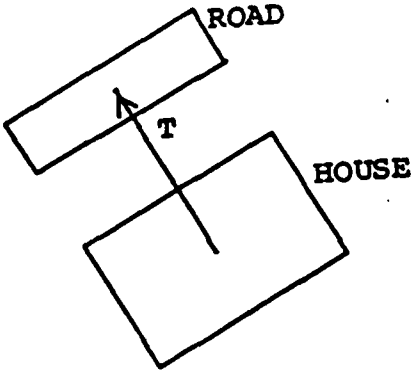


Figure 2-4. Pictorial description of house-road relation.

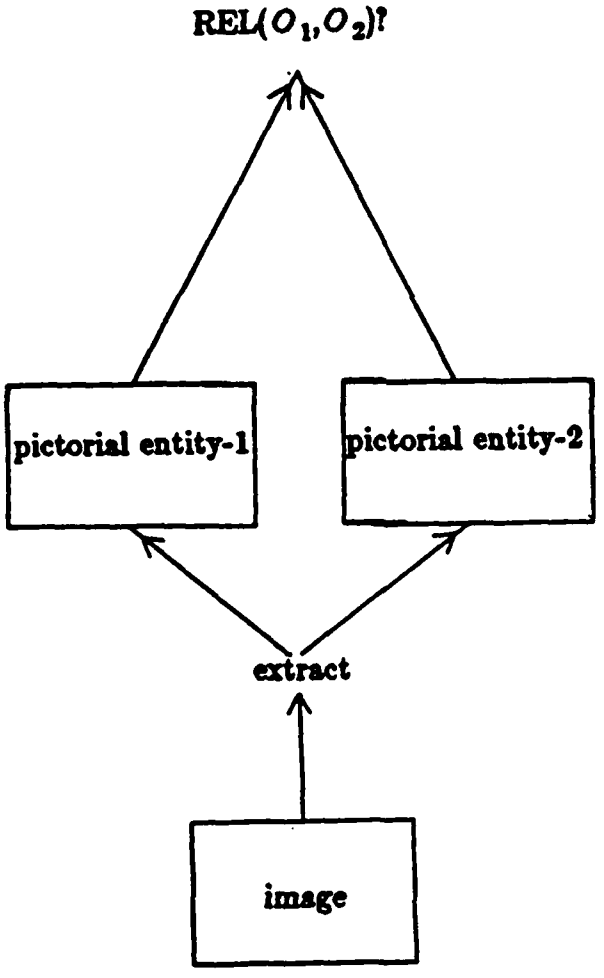
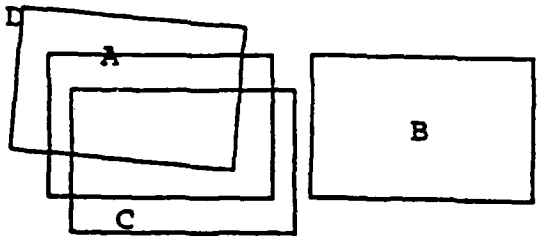
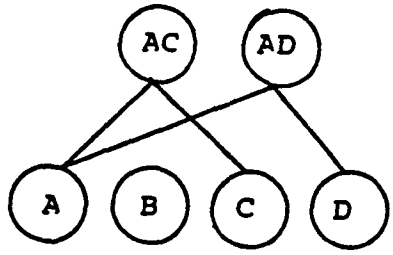


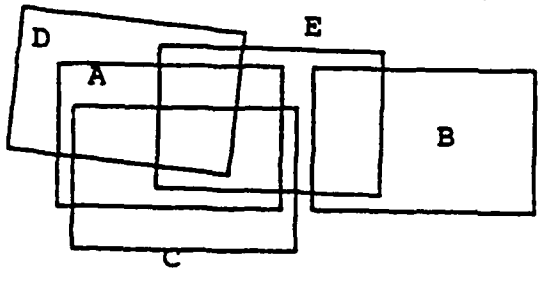
Figure 3-1. Using a relation as a constraint.

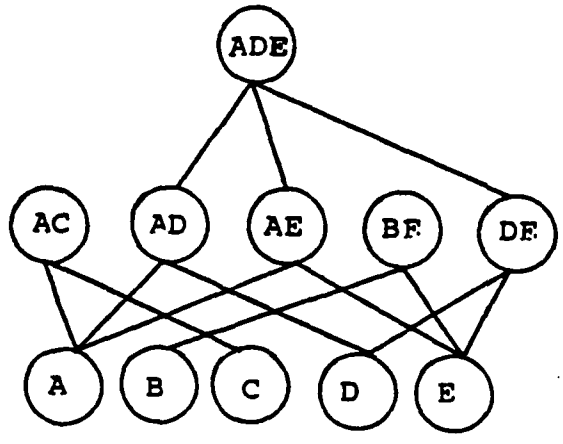




iconic descriptions **situation lattice**

Figure 3-2(a). The situation lattice before the insertion.





iconic descriptions **situation lattice**

Figure 3-2(b). The situation lattice after the insertion.

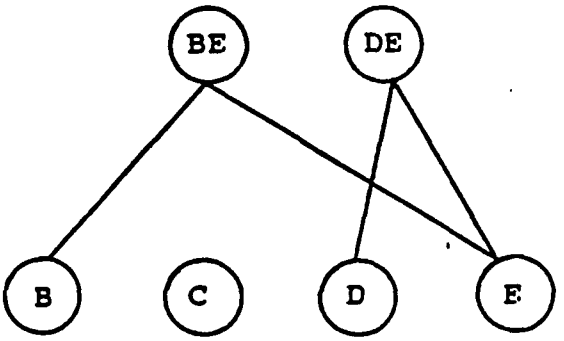


Figure 3-3. The situation lattice after the removal of A.

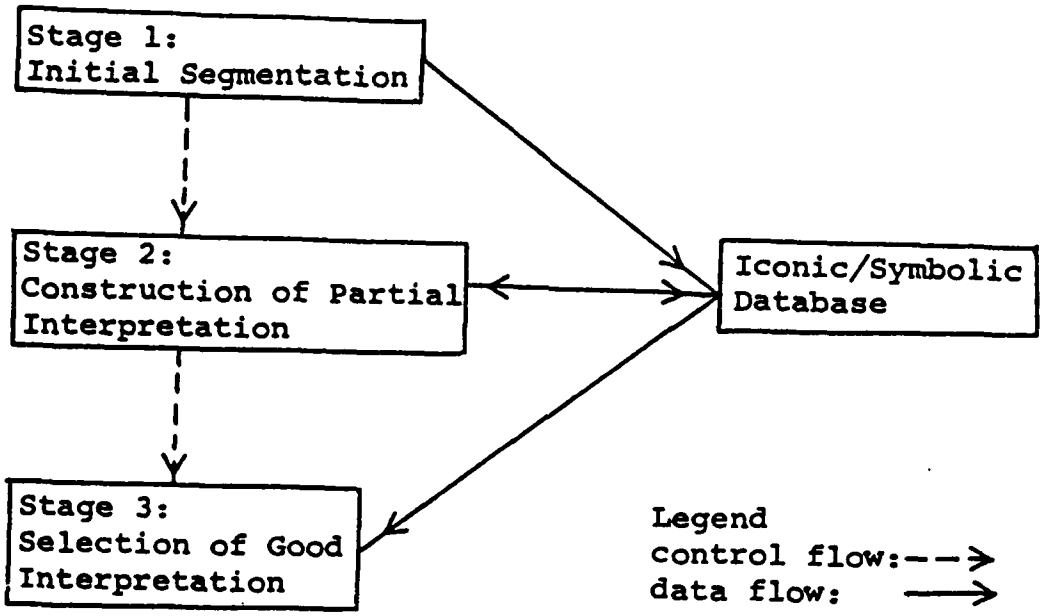


Figure 4-1. The stages of the control of SIGMA.

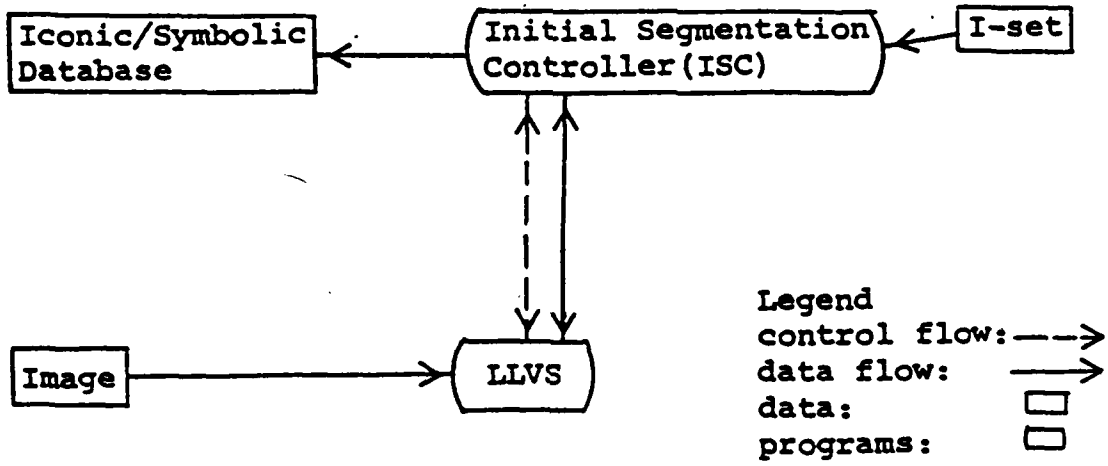


Figure 4-2. The schematic diagram of the initial segmentation process.

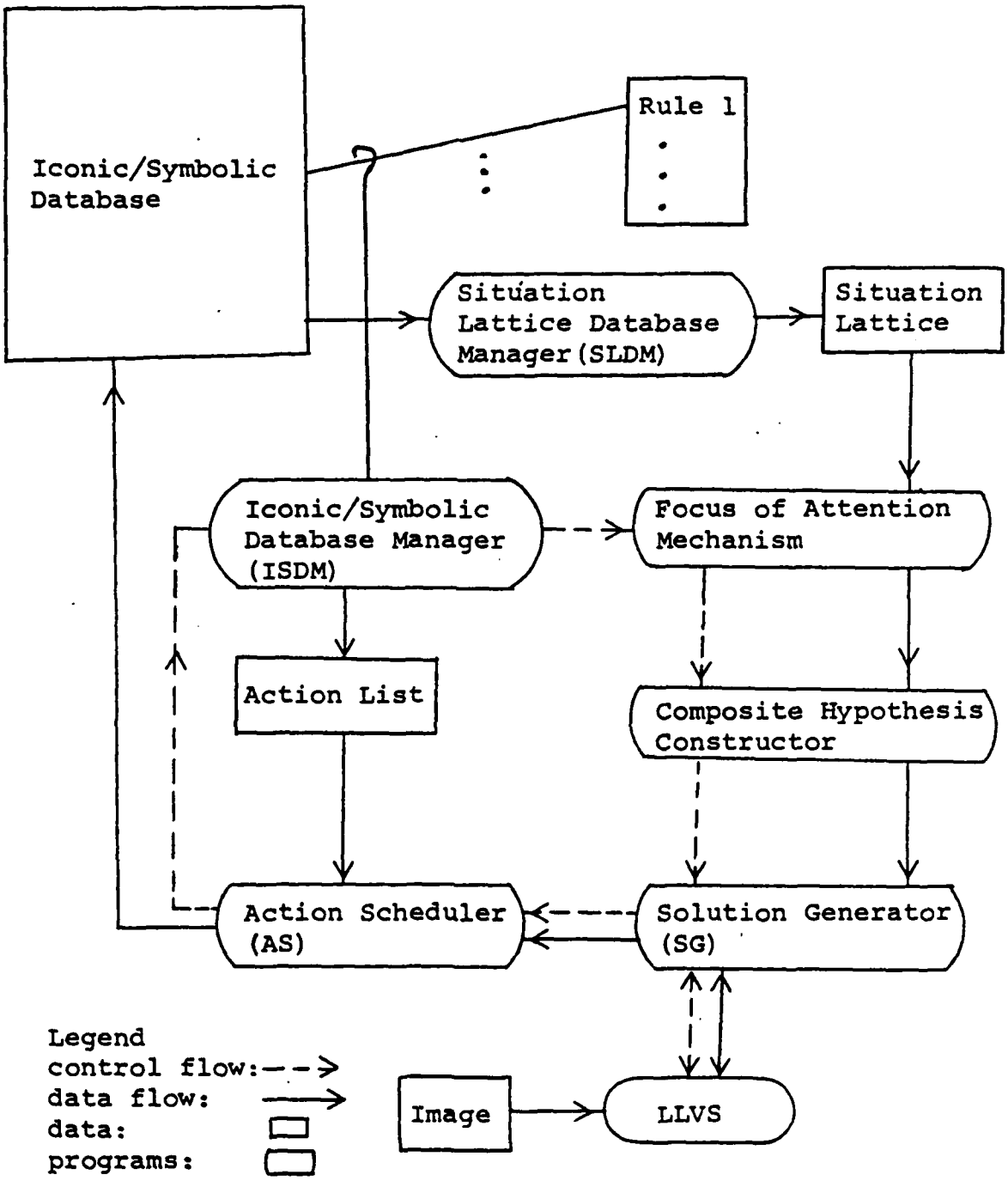
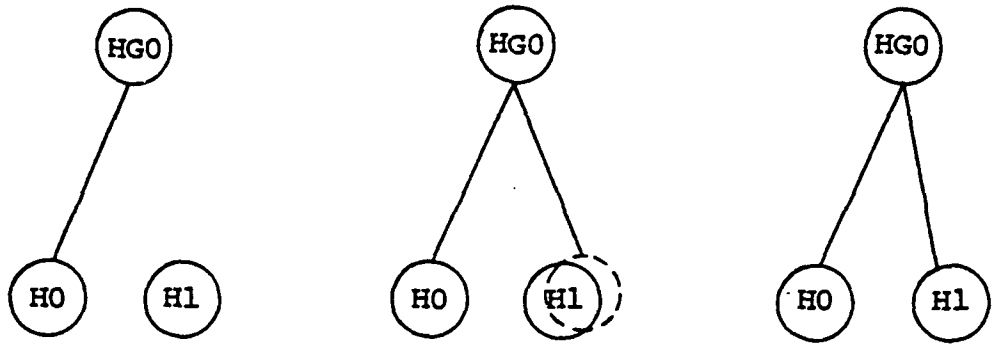


Figure 4-3. The schematic diagram of the interpretation stage.

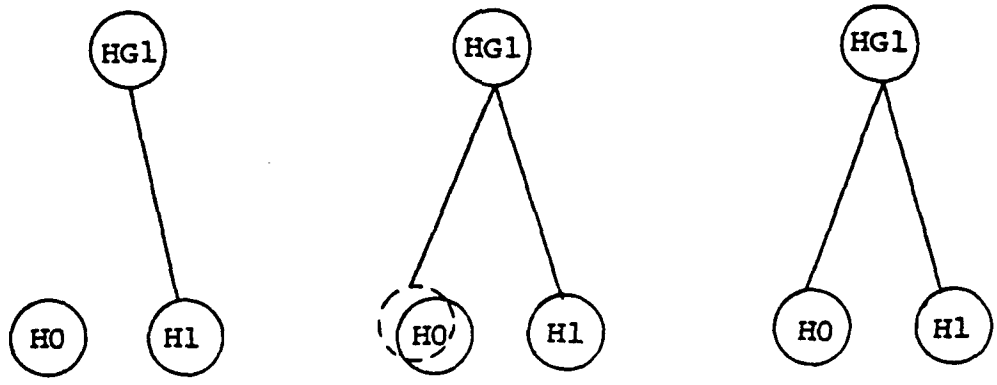


a house group instance containing H_0 is created

generate hypothesis about possible house in HG_0

fill H_1 in instance HG_0

Figure 4-4(a). Reasoning steps for constructing HG_0.



a house group instance containing H_1 is created

generate hypothesis about possible house in HG_1

fill H_0 in instance HG_1

Figure 4-4(b). Reasoning steps for constructing HG_1.

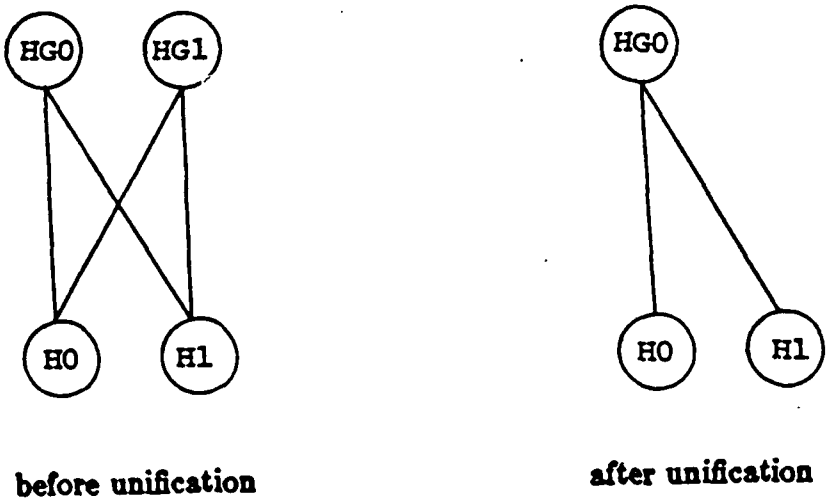


Figure 4-5. Unification of identical instances.

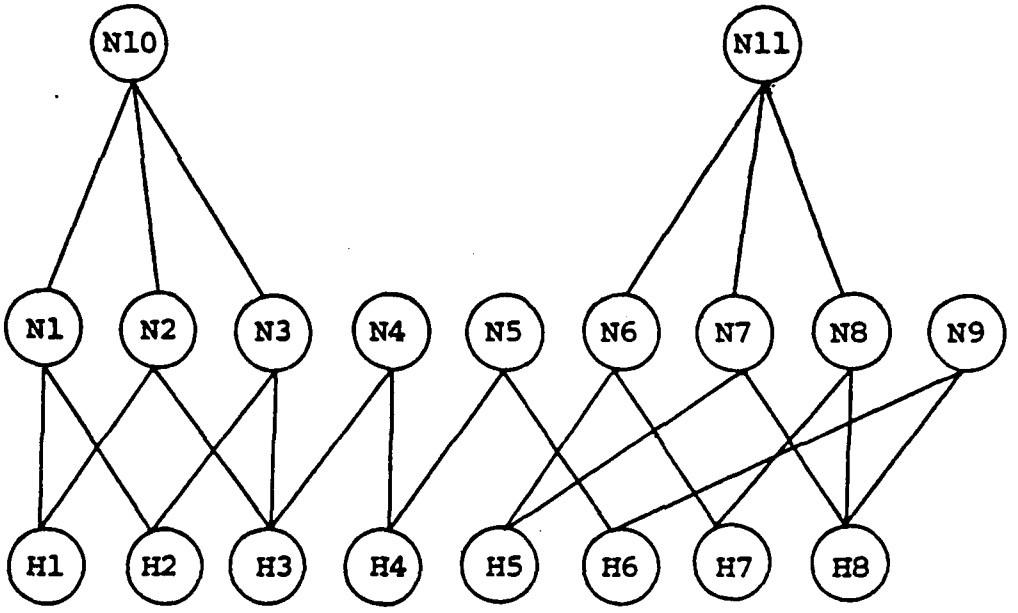
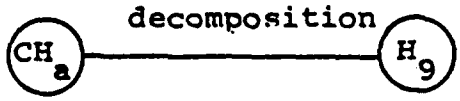


Figure 4-6. A situation lattice.



Target object of CH_a:
RECTANGULAR-HOUSE

Target object of H_9:
RECTANGLE

Delayed-action:
if* H *=nil then conclude(nil)
else conclude(make-instance(RECTANGLE-HOUSE,H)).

Figure 4-7. Decomposition of CH_a.

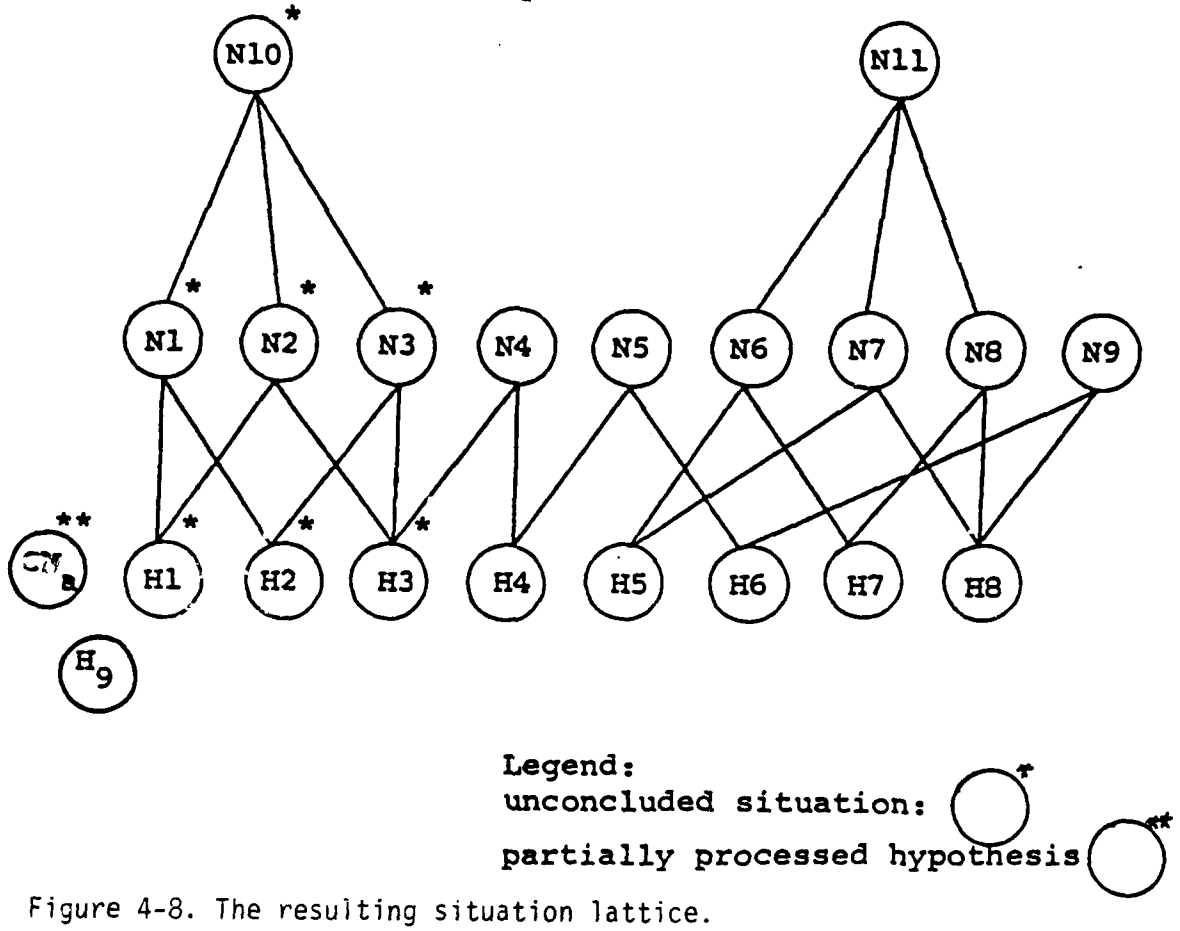


Figure 4-8. The resulting situation lattice.

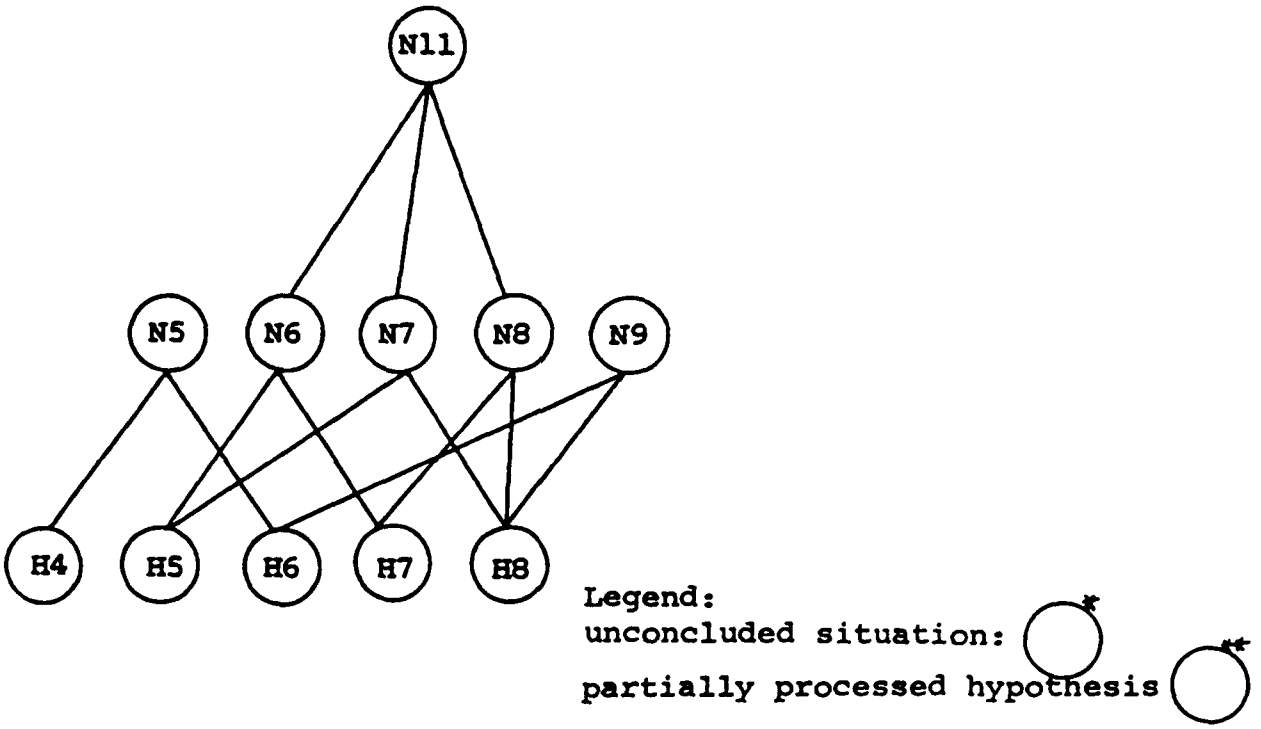


Figure 4-9. The situation lattice after actions are evaluated.

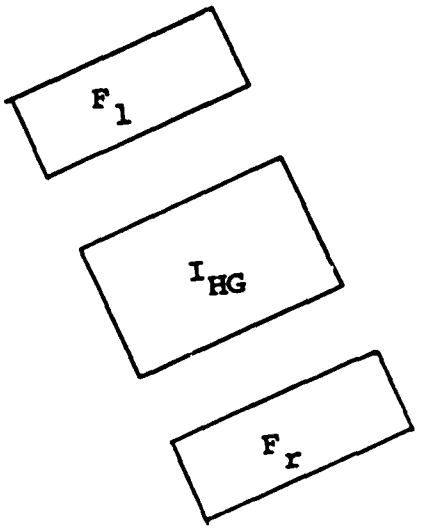


Figure 4-10. Possible road locations along I_{HG}.

Figure 5-1. An aerial image.

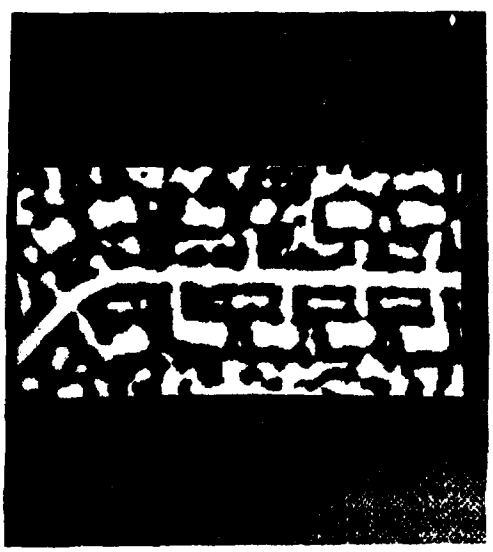


Figure 5-2. Positive connected regions.

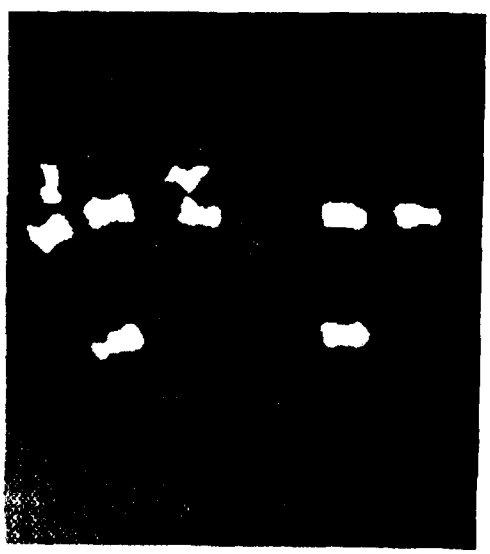


Figure 5-3. Blobs extracted by Blob-finder.

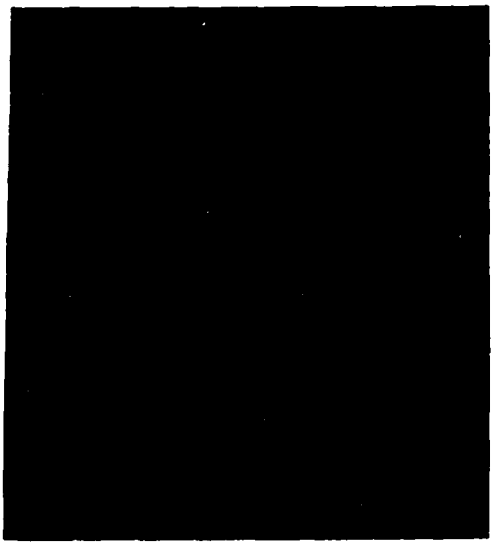


Figure 5-4. Skeletons of the connected components.

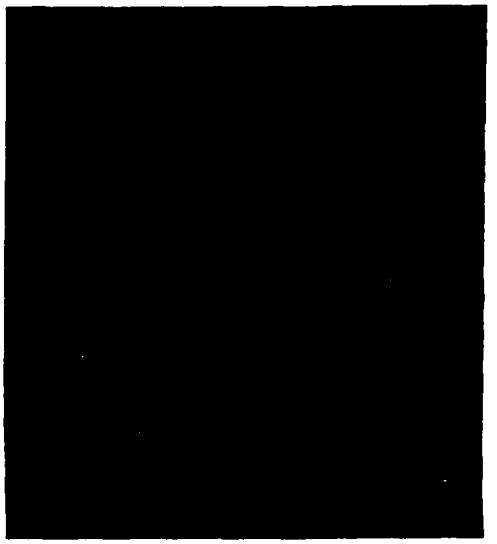


Figure 5-5. Skeletons of the ribbons extracted by Ribbon-finder.

Figure 5-6. Iconic descriptions of the RECTANGLE instances generated based on the initial segmentation process.

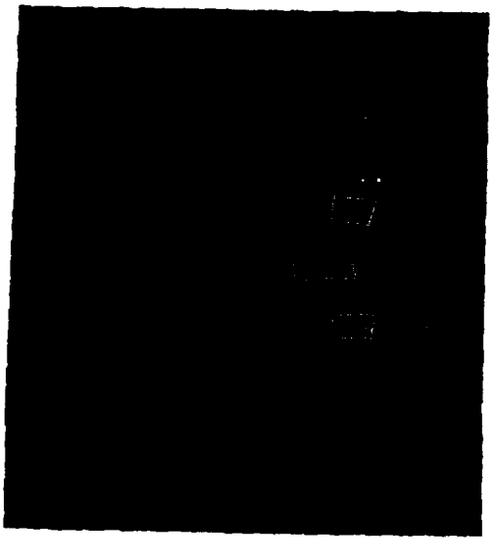


Figure 5-7(a). An example (see Section 5.3.1.)

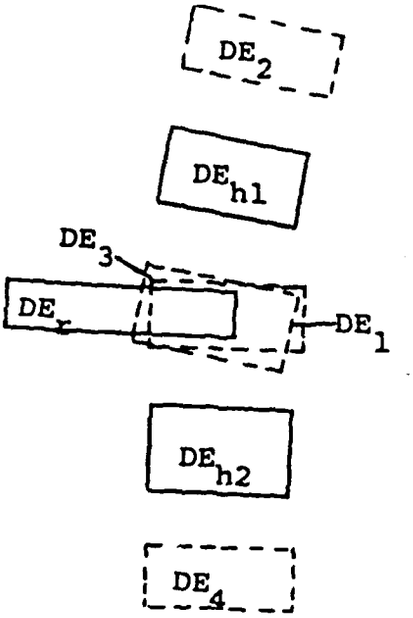


Figure 5-7(b). A depiction of the situation.

DE's	Type	Generated-by
DE_r	ROAD instance	
DE_{h1}	HOUSE-GROUP instance	
DE_{h2}	HOUSE-GROUP instance	
DE_1	ROAD hypothesis	DE_{h1}
DE_2	ROAD hypothesis	DE_{h1}
DE_3	ROAD hypothesis	DE_{h2}
DE_4	ROAD hypothesis	DE_{h2}

Table 5-1. The descriptions of the DE's.

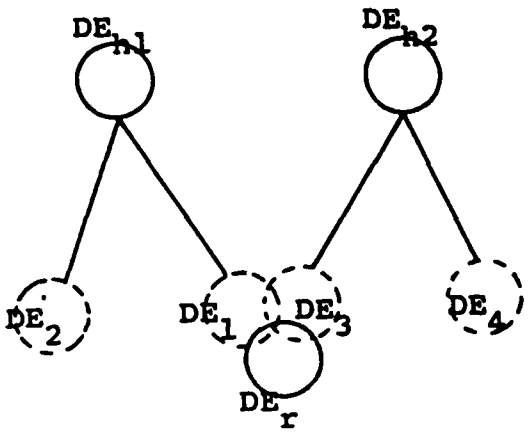


Figure 5-8. Portion of the interpretation network related to the situation.

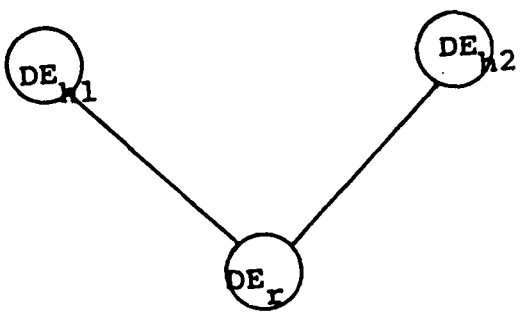


Figure 5-9. Resulting interpretation network.

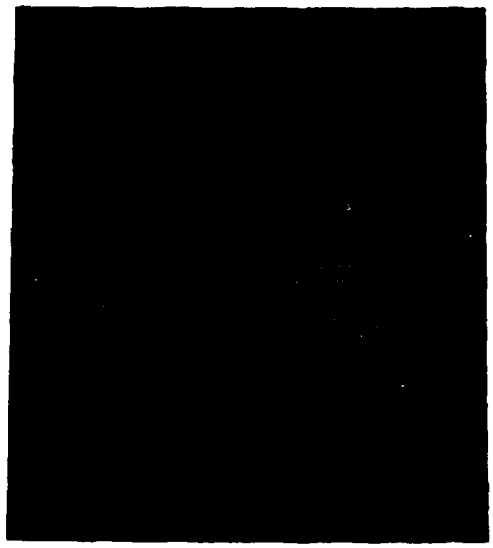


Figure 5-10(a). An example (see Section 5.3.2.)

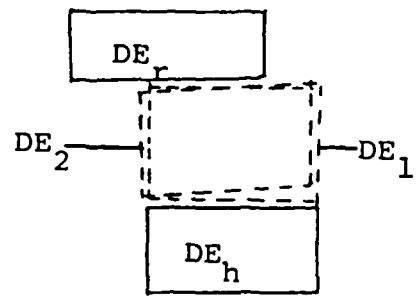


Figure 5-10(b). A depiction of the situation.

DE's	Type	Generated-by
DE_r	ROAD instance	
DE_h	HOUSE instance	
DE_1	DRIVEWAY hypothesis	DE_h
DE_2	DRIVEWAY hypothesis	DE_r

Table 5-2. The descriptions of the DE's.

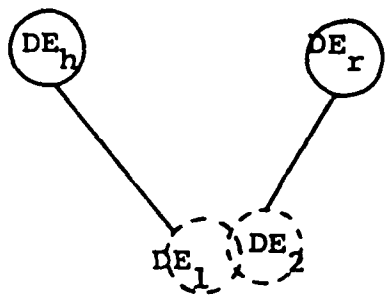


Figure 5-11. Portion of the interpretation related to the situation.

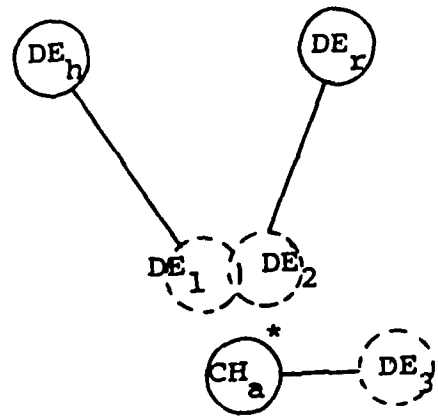


Figure 5-12. Resulting interpretation network.

Action	Cause-of-delay
$A_{first-order-properties}$	DE_3

Unconcluded-situation	Composite hypothesis
S_1	CH_a

Table 5-3. Relations between the DE's, action $A_{first-order-properties}$, and S_1.

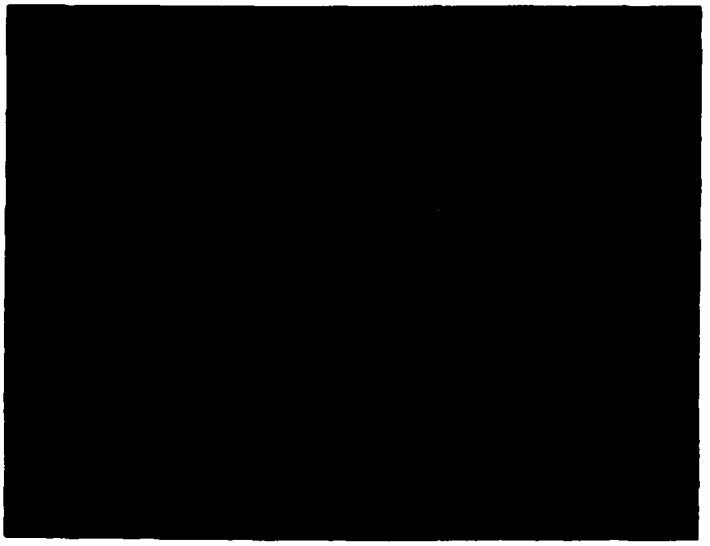


Figure 5-13. A window generated by the HLVS.

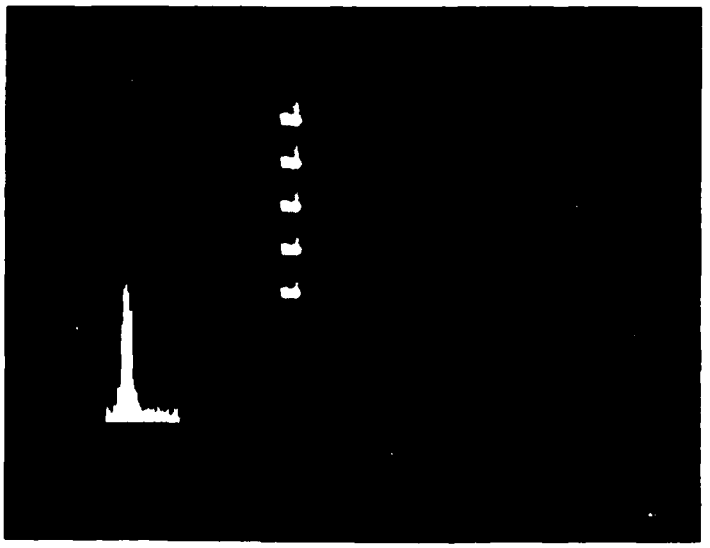


Figure 5-14. Intermediate results of the LLVS.

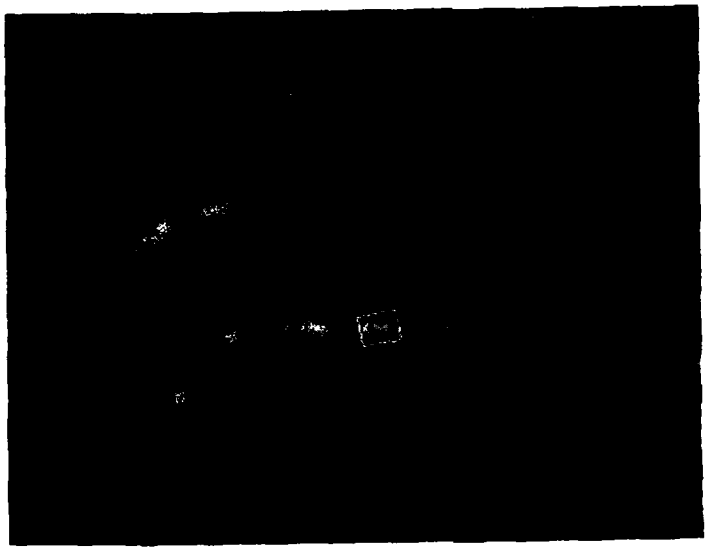


Figure 5-15. The RECTANGLE instance generated by the HLVS (based on the results computed by the LLVS).

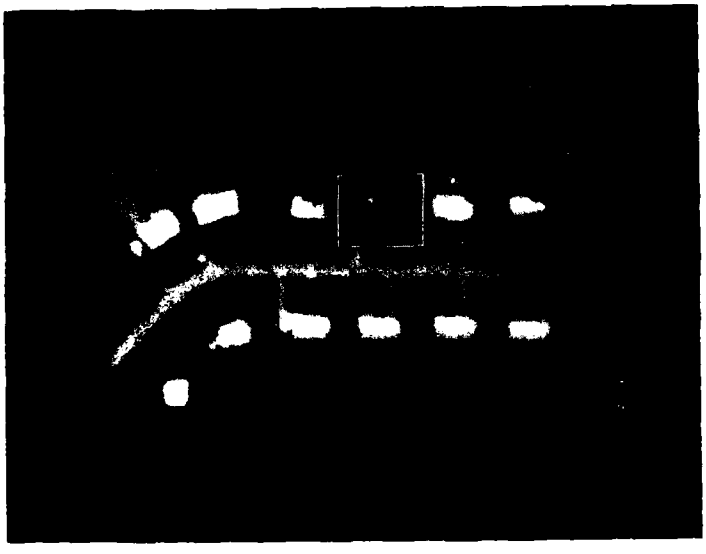


Figure 5-16. Another window generated by the HLVS.

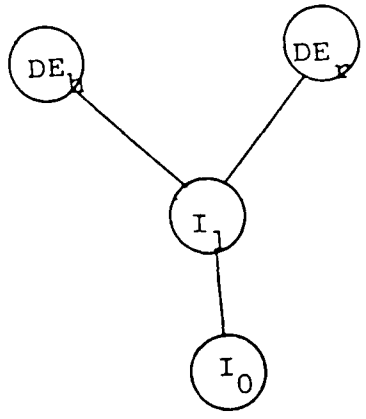


Figure 5-17. Resulting interpretation network (when a solution has been generated).

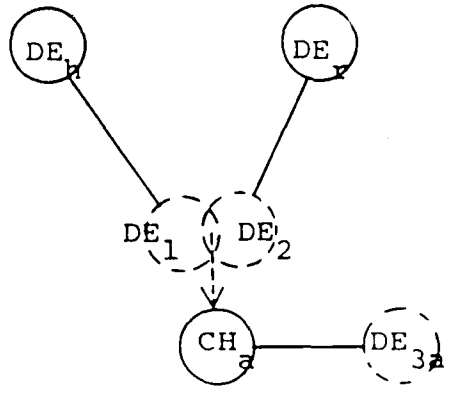


Figure 5-18. Resulting interpretation network (when no solution has been computed).

Figure 5-19. Initial set of RECTANGULAR-HOUSE instances.

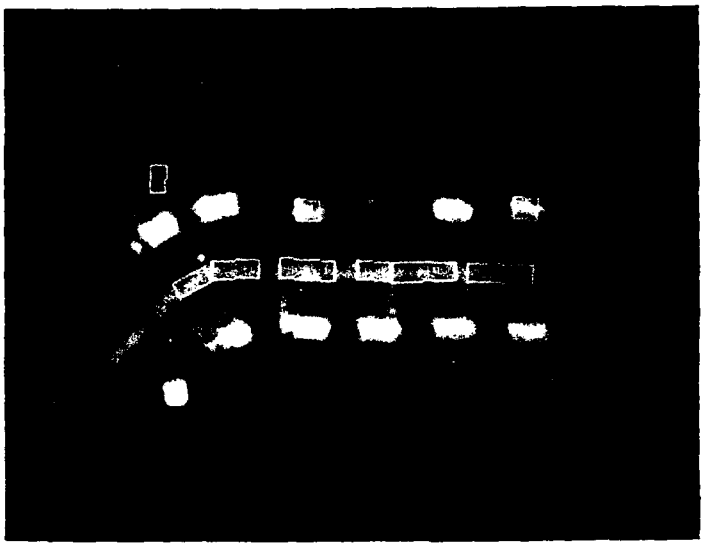


Figure 5-20. Initial set of VISIBLE-ROAD-PIECE instances.

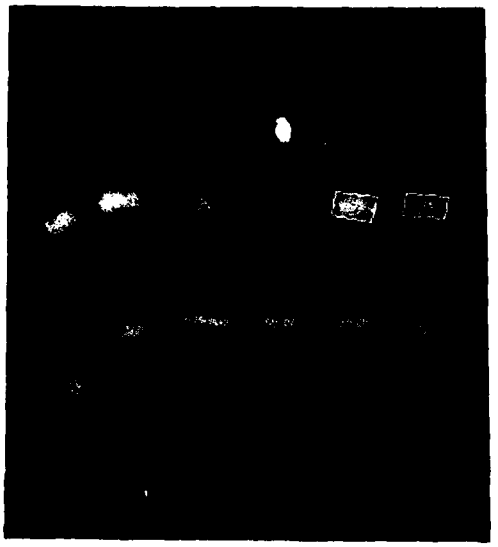


Figure 5-21(a). Two HOUSE-GROUP instances (see Section 5-4).

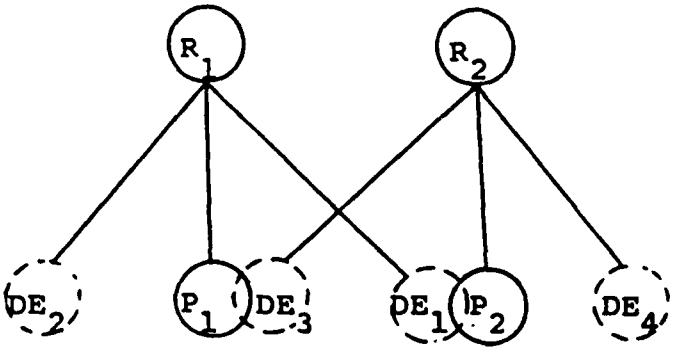


Figure 5-21(b). Portion of the interpretation network related to the situation.

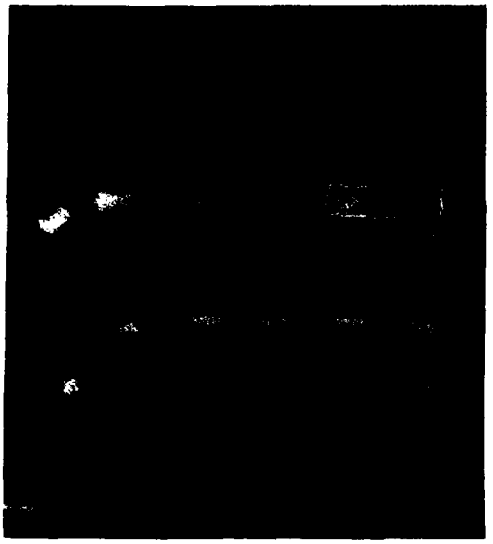


Figure 5-22(a) Resulting HOUSE-GROUP instance R_1.

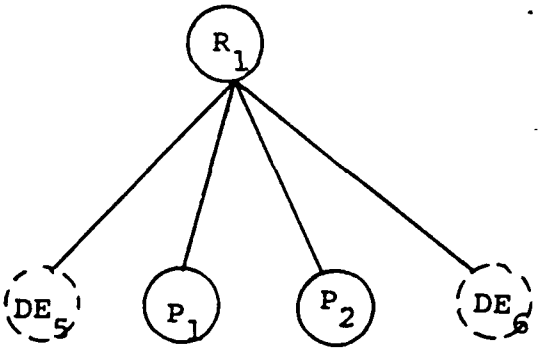


Figure 5-22(b). Hypotheses generated by R_1.

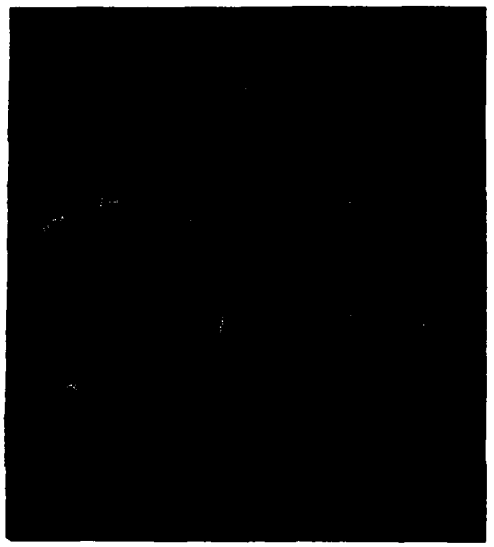


Figure 5-23(a). Two HOUSE-GROUP instances (see Section 5-4).

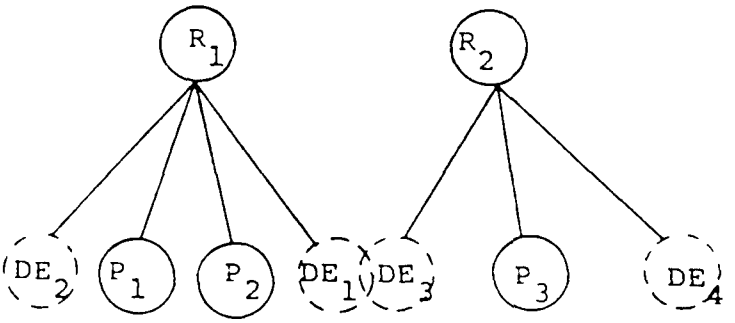
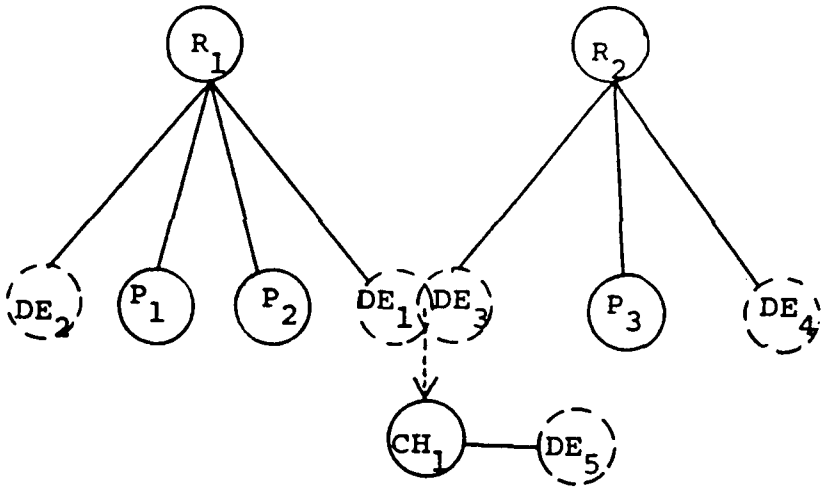
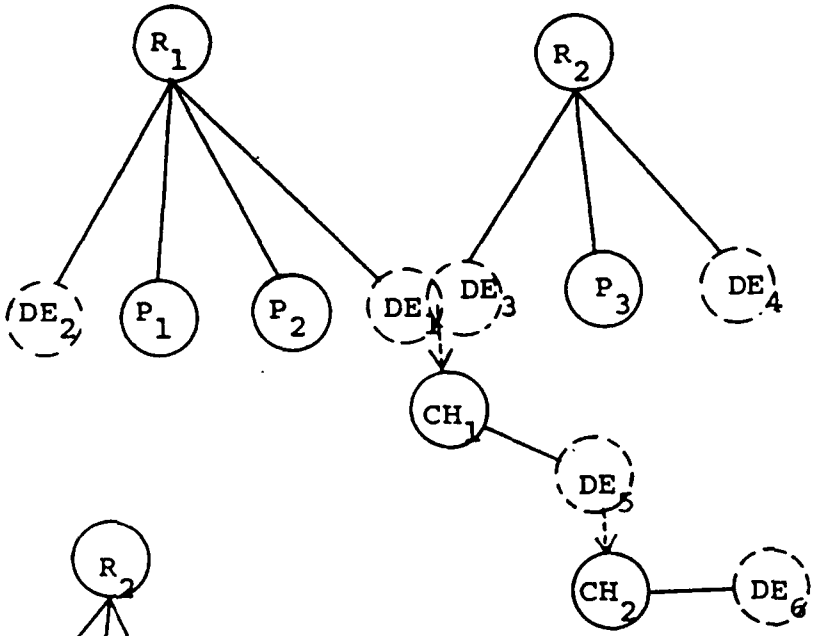


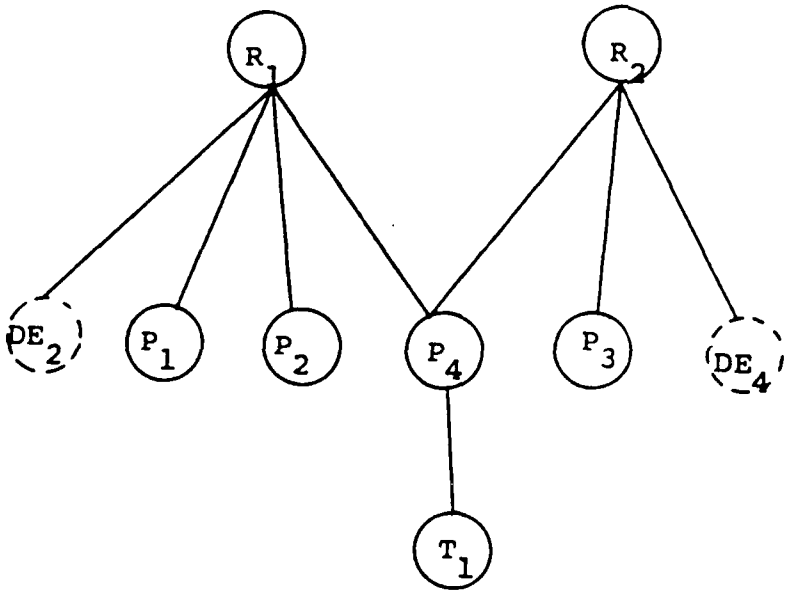
Figure 5-23(b). Portion of the interpretation network related to the situation.



5-24(a).

5-24(b) --



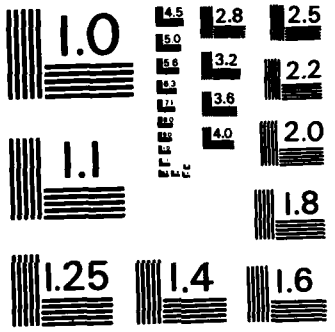


5-24(c).

Figure 5-24. Snapshots of the interpretation network related to R_1 and R_2 (see Figure 5-23) at various stages of the processing.

AD-A160 129 HYPOTHESIS INTEGRATION IN IMAGE UNDERSTANDING SYSTEMS (U) MARYLAND UNIV COLLEGE PARK CENTER FOR AUTOMATION RESEARCH V S HWANG ET AL. JUN 85 CAR-TR-130 2/2

UNCLASSIFIED AFOSR-TR-85-0876 F49620-83-C-0082 F/G 14/5 NL



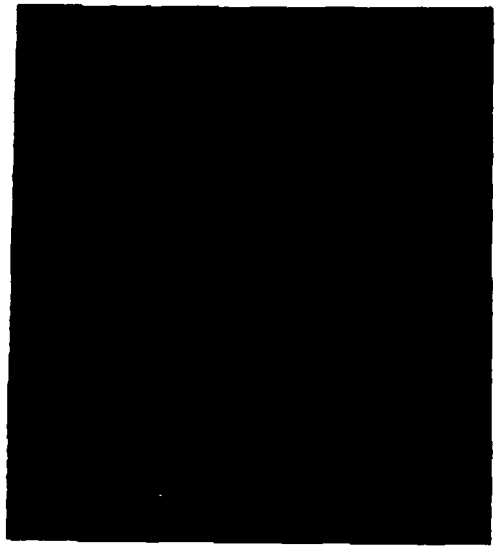


Figure 5-25(a). Resulting HOUSE-GROUP instance.

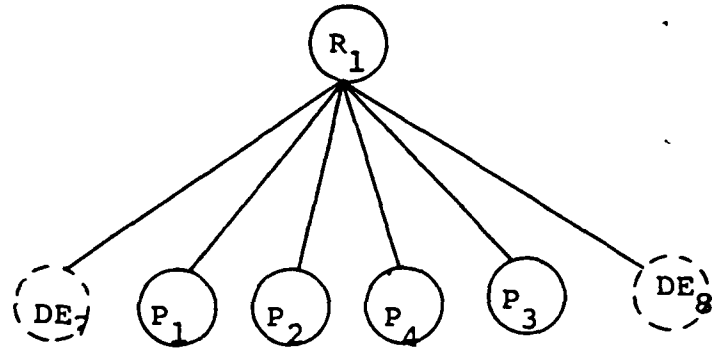


Figure 5-25(b) Resulting interpretation network.

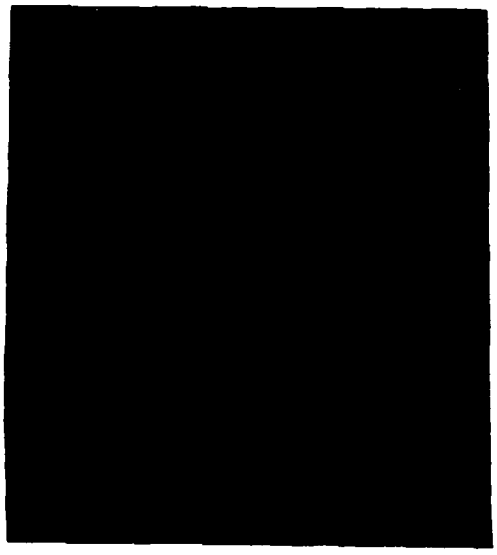


Figure 5-26(a). Two ROAD instances (see Section 5-4).

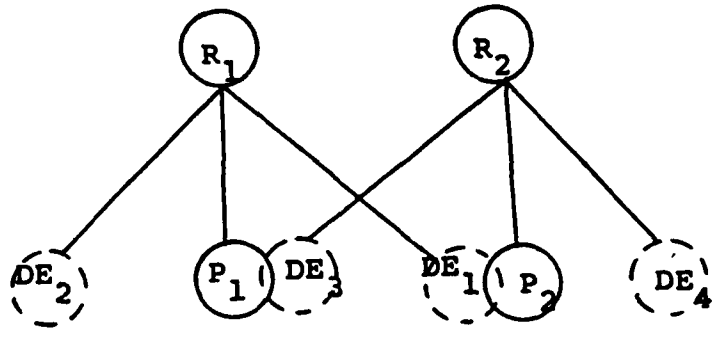


Figure 5-26(b). Portion of the interpretation network related to the situation.

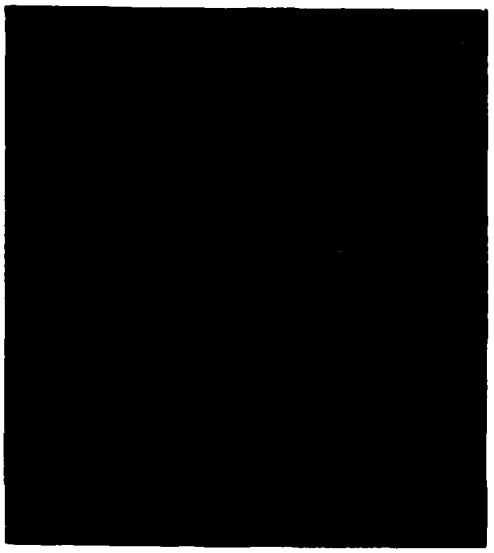


Figure 5-27(a). Resulting ROAD instance.

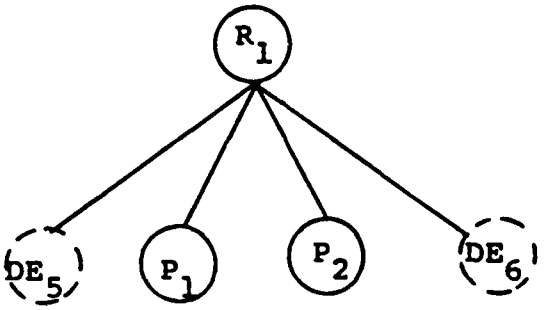


Figure 5-27(b). Resulting interpretation network.

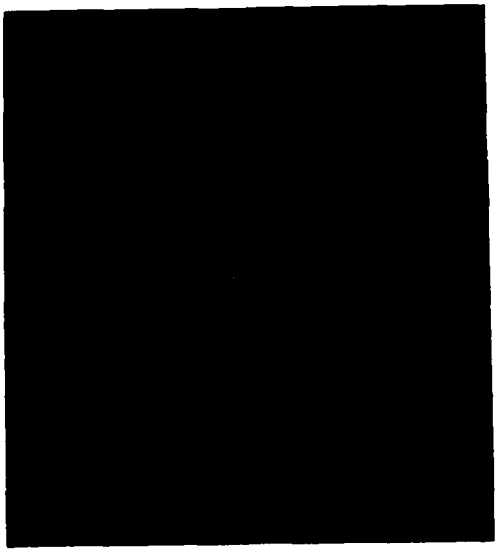


Figure 5-28(a). Two ROAD instances (see Section 5-4).

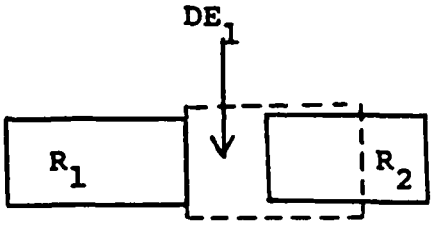


Figure 5-28(b). A depiction of the situation.

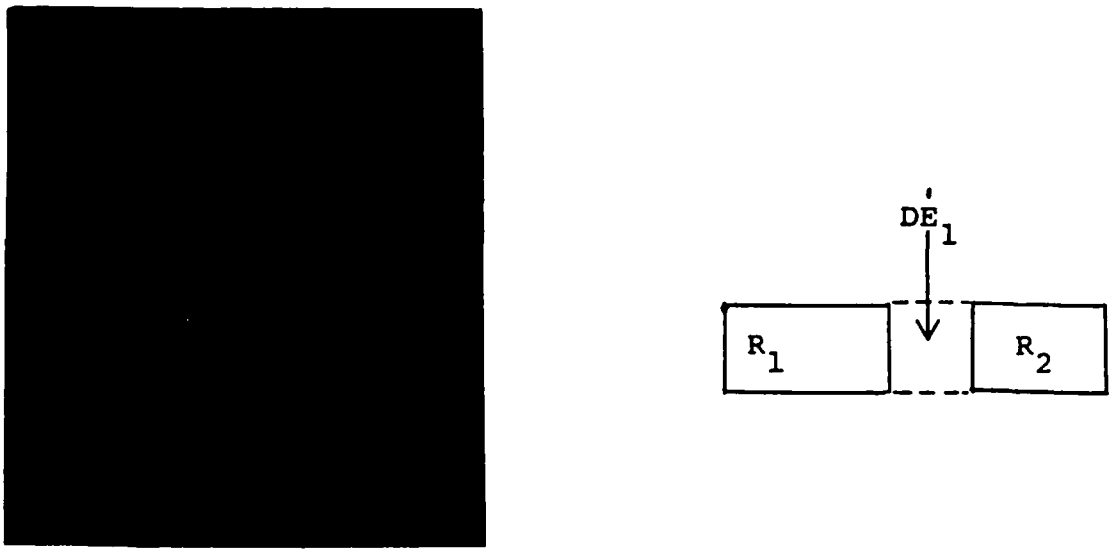


Figure 5-29. Hypothesis DE_1 has been modified.

Figure 5-30. A ROAD-TERMINATOR hypothesis has been generated.

Figure 5-31. Iconic description of a situation and its interpretation network (see Section 5-4).

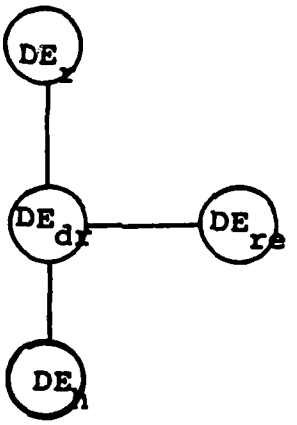


Figure 5-32. Resulting interpretation network.

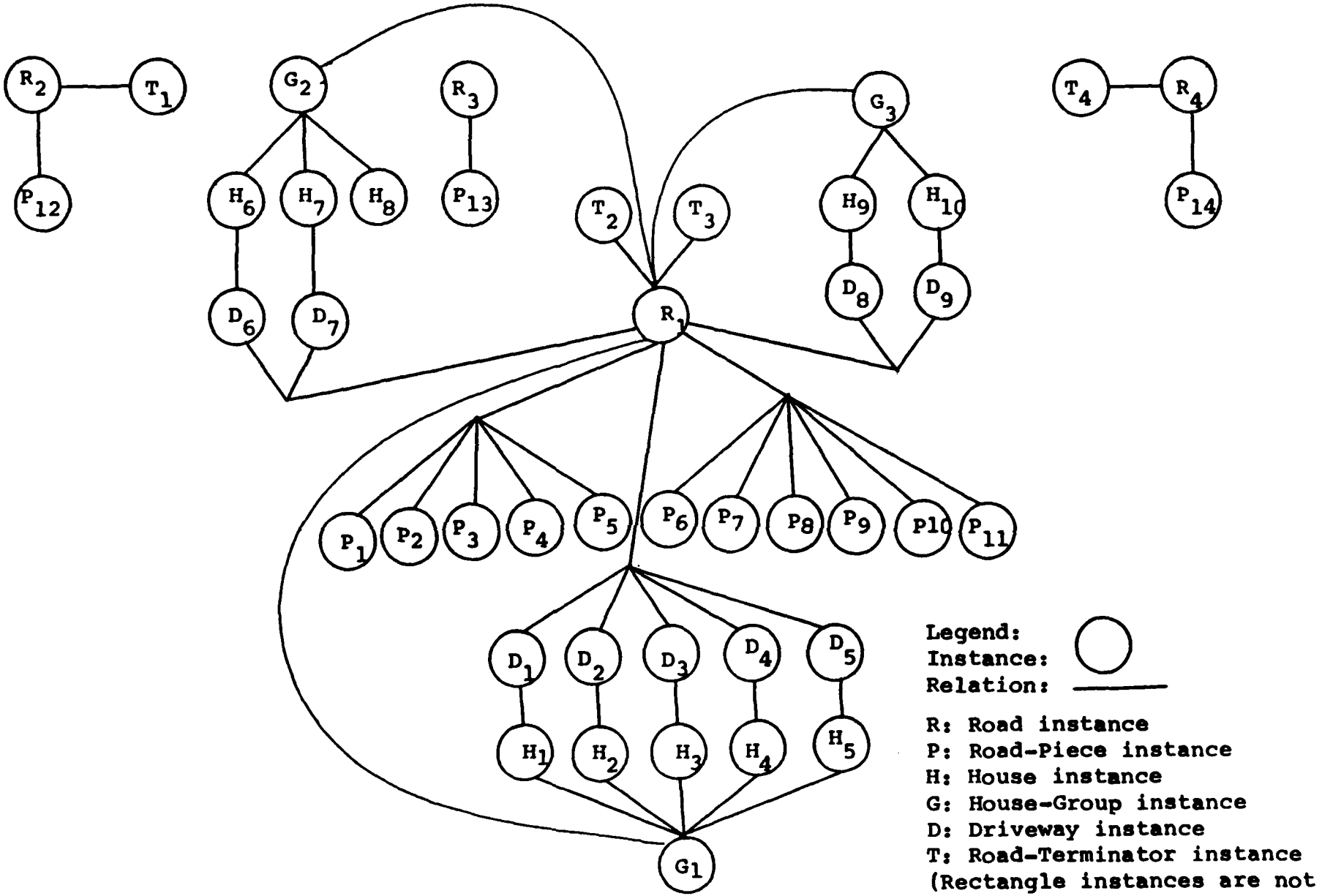


Figure 5-33. Final interpretation network.

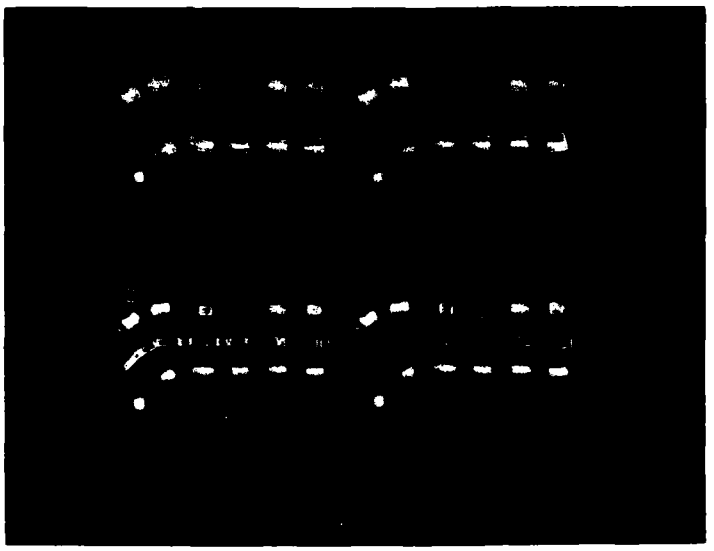


Figure 5-34. Final results.

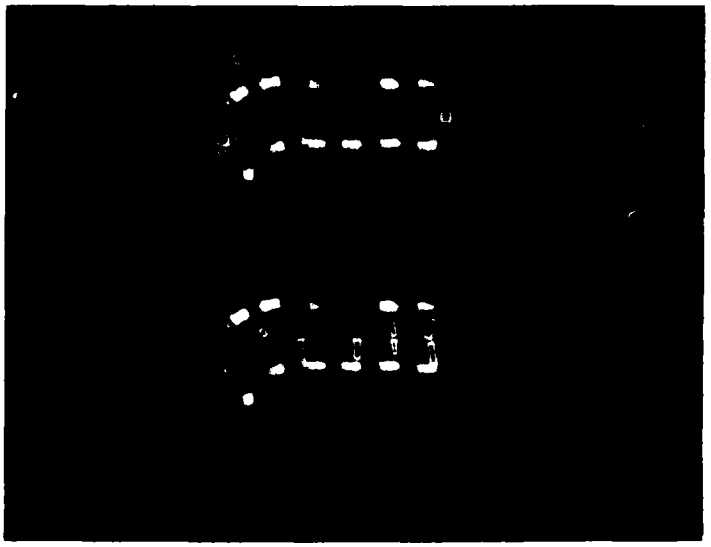


Figure 5-35. Final results (cont.).

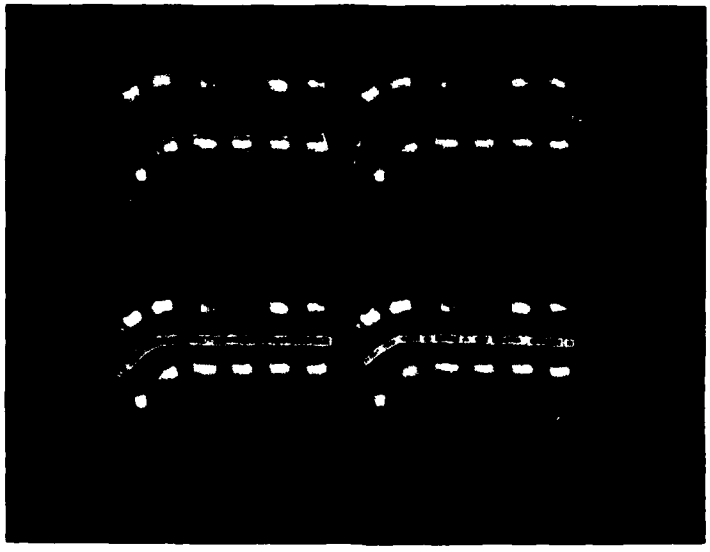


Figure 5-36. Explanation of a ROAD instance.

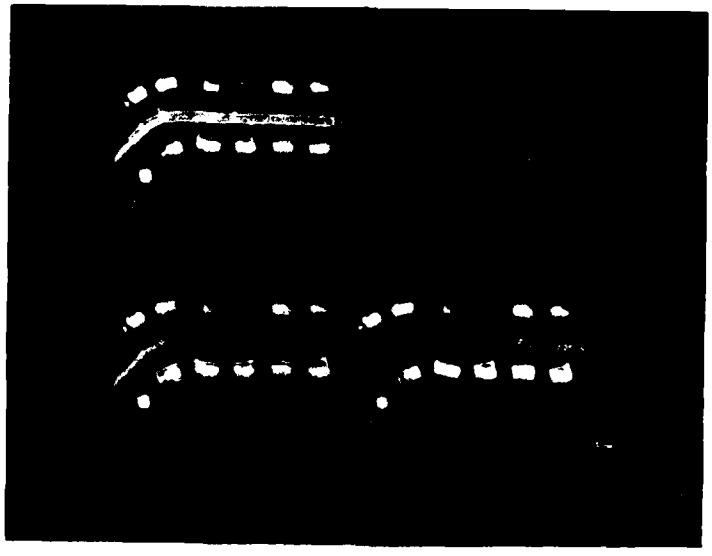


Figure 5-37. Explanation of a HOUSE GROUP instance.

END

FILMED

11-85

DTIC